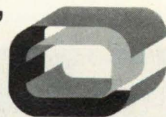
Bit
UNA PUBBLICAZIONE
DEL GRUPPO EDITORIALE JACKSON
ANNO 6 N. 40 GIUGNO
LA PRIMA
E PIU' DIFFUSA RIVISTA
DI PERSONAL COMPUTER
L. 4.000
BITEST
ICL
PERSONAL
COMPUTER
IL MICRO
A PORTATA
DI MANO
SEMAFORO
CIBERNETICO
TRASMISSIONE
DATI
DBASE II
TUTTO
SULL'IEEE 488
ICL PERSONAL COMPUTER

Arrivano i nostri.

Parliamo dei computers, o, se preferisci, di un avvenimento importante per l'efficienza della tua azienda.
Oggi infatti diventa facile applicare i computers piú affidabili, garantiti dal nome delle marche che contano di piú al mondo: con queste, la tua azienda farà un volo che la porterà avanti di anni e anni.
Vieni a Computer City: una vasta rete di centri specializzati nei personal computers, i piú adeguati alle tue necessità, dove la vendita viaggia con l'assistenza di un personale esperto e qualificato, di cui ti puoi fidare.
Tutti pionieri a Computer City.

computer city
Parla la tua lingua.

Realität

Anno 6 - n° 40 - Giugno 1983

In copertina:
ICL Personal Computer
(fotografie: Photo In)

INDICE INSERZIONISTI

Sommario

DIRETTORE RESPONSABILE
Giampietro Zanga
DIRETTORE TECNICO
Gianni Giaccaglini
CAPO REDATTORE
Pietro Dell'Orco
REDAZIONE
Lorenzo Barrile
Paolo Capobussi
REDAZIONE USA
Sergio Mello-Grand
Gabriella Martino
GRAFICA, IMPAGINAZIONE E DISEGNI
Gianfranco De Rienzo
FOTO
Photo-In
DIFFUSIONE E ABBONAMENTI
Ombretta Giannetto,
Adela Bel Lozano,
Luigi De Cao, Marco Benedetti,
Claudio Bautti

GRUPPO EDITORIALE JACKSON Srl
DIREZIONE, REDAZIONE, AMMINISTRAZIONE
Via Rosellini, 12 - 20124 Milano
Telefoni 68.03.68 - 68.00.54
Telex 333436 GEJIT I
SEDE LEGALE
Via Vincenzo Monti, 15 - 20123 Milano
REDAZIONE USA
GEJ Publishing Group Inc.
1143 Quince Avenue
94087 Sunnyvale, CA
Tel. (408) 7730103
DIREZIONE EDITORIALE
Giampietro Zanga e Paolo Reina
COORDINAMENTO EDITORIALE
Daniele Comboni
Autorizzazione del Tribunale di Milano n. 445 del 16/12/1978

PUBBLICITA': Concessionario per l'Italia e l'Estero
Reina s.r.l.
Via Washington, 50 - 20146 Milano
Tel. (02) 4988066/7/8/9/060 (5 linee r.a.)
Telex 316213 REINA I

Concessionario pubblicità per USA e Canada:
International Media Marketing
16704 Marquardt Avenue
P.O Box 1217
Cerritos CA 90701
(213) 926-9552

STAMPA:
Reweba (Brescia)

Concessionario esclusivo per la diffusione in Italia e all'Estero:
SODIP - Via Zuretti, 25 - 20125 Milano

Spedizione in abbonamento Postale Gruppo III/70

Prezzo della rivista L. 4.000
Numero arretrato L. 6.000
Abbonamento annuo L. 35.000
per l'Estero L. 52.000
I versamenti vanno indirizzati a:
Gruppo Editoriale Jackson
Via Rosellini, 12 - 20124 Milano
mediante emissione di assegno bancario, cartolina vaglia o utilizzando il c/c Postale numero 11666203
Per i cambi di indirizzo, indicare, oltre naturalmente al nuovo, anche l'indirizzo precedente, ed allegare alla comunicazione l'importo di L. 500, anche in francobolli.

Mensile associato all'USPI Unione Stampa Periodica Italiana

Diecinque

DIECINQUE, proseguendo nel proporre il meglio del mercato dell'informatica, accanto alla affermata serie dei sistemi ABC, simbolo della più alta tecnologia ed efficienza nella gestione aziendale, implementa ed espande la gamma dei suoi prodotti introducendo la nuova linea di super microcomputer a 16 Bit.

Questa nuova famiglia si basa su una struttura con più microprocessori a 16 Bit della famiglia INTEL.
La memoria principale parte da 512KB per espandersi fino ad 1MB mentre le memorie di massa collegabili sfruttano le più moderne tecnologie a Winchester per unità multiple.
Sono inoltre disponibili 16 linee per collegare più unità video-terminali, monocromatici o a colori con possibilità grafiche di grande rilievo, unità stampanti e periferiche specializzate.
Tutto ciò è supportato da vari Sistemi Operativi quali il GENIX (versione AI del noto Sistema Operativo UNIX™), nonché il CP/M 86, MP/M86, MS DOS.

DIECINQUE integra i componenti, Hardware, Software di Base e Software Applicativo, con un servizio di formazione professionale e assistenza tecnica altamente qualificato.

Diecinque

L'informatica presente e futura.

DIECINQUE S.p.A. 40069 Zola Predosa - Bologna
Via Nannetti, 1 - Tel. (051) 752264 - Telex 214848 OLCIN I

Ai Electronics Corp.
Tokyo-Japan.

Editoriale

Tecnologia del desiderio?

La pervasività della microinformatica non manca di suscitare in modo ricorrente gli allarmi o le ironie, più o meno gravi, delle Cassandre moderne. Questi disastrografi disegnano prospettive inquietanti, in fondo alle quali c'è il soggiogamento integrale dell'individuo, avvinto al terminale come un Charlie Chaplin alla catena di montaggio, costretto a dar in pasto al computer tutti i dati di cui questo Moloch incessantemente si pasce, insaziabile e del tutto irrispettoso d'ogni più sacrosanta privacy. E via con la lagna cupa del medioevo-prossimo-venturo. Non serve a costoro che utoposti di diametrale estremismo, come Jean Jacques Servan Schreiber e il geniale Seymour Papert (a proposito: han divorziato? si sono riappacificati? se qualcuno, nel Villaggio Informatico, ne sa qualcosa faccia un telefischio ...) delineino invece le tinte rosate ed idilliache d'un mondo tutto progresso e gaudio, dall'inquinamento ridotto (grazie alla telelaborazione che annulla o quas pendolarismi e traffico) e dal lavoro-zero-salario-intero (grazie ai robot ed ai personal): per cui non resta che starsene in casa a giocare con la "turtle" o a robotwar. Beh, non esageriamo: c'è pure la pesca.

Del tutto dimentichi che l'orwelliano 1984 è bell'e arrivato senza che il Grande Fratello si sia fatto vedere, anzi appare sempre in più netto ribasso l'informatica centralizzata a favore di quella distribuita e, semmai, caotica ed un poco pasticciona, i novelli piagnoni incalzano profilando l'avvento dei prodotti d'intelligenza artificiale come estrema spoliazione dell'animo umano stesso. E giungono, i tremebondi, a valutare come un terribile segno premonitore di questo "portar all'ammasso il cervello" l'idolatria di Time che, non l'avesse mai fatto, ha proclamato il personal come uomo dell'anno. Qui c'è inserito molto bene John Kirkley, il notista di Datamation a smitizzare questo evento, mediante un azzeccato parallelo con il riconoscimento conferito all'attore maschio Dustin Hoffman come migliore attrice, per "Tootsie": per essere detta da un appartenente alla truce Era Informatica è molto spiritosa, non vi pare?

Il dubbio semmai, per chi segue il mercato dei personal, è se questi oggetti non abbiano al contrario un'eccessiva natura giocherellona-consumistica. Ci riferiamo alla fascia bassa, perchè la nuova generazione ormai aspira decisamente ad uno status ultraserio, al punto che taluni (ciavete fatto caso?) si autoproclamano "professional", gli sdegnosi. Ma per gli (allegri) spettri VIC il fatto in qualche modo sussiste: basti pensare alla riproposta di attività ricreative coi personal presso i Club Mediterranee, con tutto il rispetto che, tutto sommato, l'iniziativa merita.

Lungi da noi però ogni idea di scandalizzarsi per questo aspetto consumatoriale, specie in questo frangente in cui sogniamo d'andare in ferie pure noi (ma, forse senza portarci appresso il p.c.).

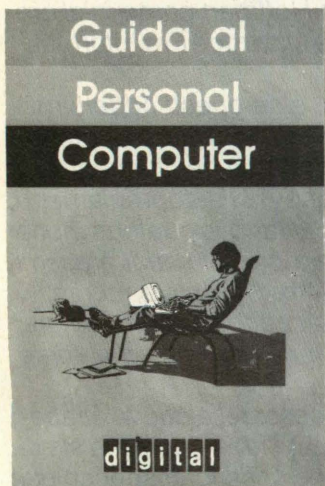
Visto che il fenomeno in fondo favorisce l'auspicato processo della nuova alfabetizzazione, solo agli aristò vetero-informatici può dar fastidio l'idea che essa sia di massa. E se i giovani vengono allettati con i pur subdoli e sofisticati specchietti della tecnologia del desiderio rimane il fatto, senz'altro positivo, che essi sono animati dal desiderio della tecnologia. Per dominarla, almeno si spera.

Gianni Giaccaglini

Galassie e gigantopianeti: l'uva non è matura

Con i nuovi chip ad accoppiamento di carica (charge couple device) prodotti dalla Gould si sta sviluppando repidamente il settore dell'elaborazione di immagini. Un CCD è costituito da una sottile membrana di silicio spessa 1/100 di millimetro. La sua sensibilità alla luce è dieci volte maggiore di quella di una emulsione fotografica. Il suo impiego per fotografare il cielo sta portando una piccola rivoluzione nel campo dell'astronomia. Infatti molti oggetti celesti che emettono luce debolissima possono venire studiati meglio che in precedenza. Tuttavia anziché portare soluzione alle vecchie domande, la nuova apparecchiatura ne suscita delle nuove che rimangono senza alcuna risposta. Recentemente si sono ossevati mediante l'uso dei CCD "getti" di materia vicino al nucleo di certe galassie. L'emissione luminosa di questi "oggetti" pare non obbedisca ad alcuna delle leggi matematiche fin qui applicate dagli astronomi. Un altro campo d'indagine con l'elaborazione di immagini realizzabile con i CCD riguarda i quasar. Questi corpi celesti misteriosi (il loro nome è una abbreviazione di "quasi-star") sono una vera fonte di disperazione per gli astronomi. Per un po' si è creduto che fossero gli oggetti più antichi dell'universo, basandosi sul fatto che il loro red-shift era esageratamente elevato. Ma la teoria, come tutte, non appare convincente più di tanto. Ipotesi suggestive quanto arbitrarie vengono avanzate. Aldilà delle teorie sono però curiosi gli effetti luminosi che i quastar producono. Uno è l'effetto di lente gravitazionale in base al quale si vedono sulla terra due immagini distinte dello stesso quastar. Un secondo campo di investigazione con i CCD riguarda i giganti gassosi del nostro sistema solare Giove e Saturno. Si può studiare la struttura della loro atmosfera elaborando la luce riflessa a diverse lunghezze d'onda. Il risultato di questo lavoro è però abbastanza sconfortante: non siamo in grado di comprendere le cose che vediamo. Questo ci aiuta a riflettere sulla potenza della mente che ha progettato tutto ciò.

Una guida gratuita

La Digital Equipment Corporation ha editato e mette a disposizione di tutti coloro che lo richiedono una "guida al personal computer". Si tratta di un libretto in brossura di sette capitoletti che illustrano che cosa sia un personal, che cosa ci si possa fare, e cosa fanno i personal della Digital stessa. Lo si può ottenere richiedendolo a:

Digital Equipment SpA
Servizio Comunicazioni di Marketing
V.le Fulvio Testi, 11
20092 Cinisello B. (MI)

I quattro della Televideo

La Televideo Systems Inc. ha annunciato, attraverso il suo fondatore e presidente Phillip Hwang, la disponibilità di quattro nuove unità. Due di queste sono microcomputer a 16 bit, mentre due sono a 8 bit. Tutte quattro le macchine sono basate sul sistema operativo CP/M e sono previste per essere integrate in reti di dati locali. In questo senso il sig. Hwang si dichiara persuaso che l'economicità dell'approccio alla multifunzione e multiutenza ottenibile mediante una delle quattro nuove unità presentate, non mancherà di produrre nuovi standard di prezzi nel settore dei microcomputer professionali. I nuovi modelli che sono denominati TS 1603 e TS 1600 per i sedici bit (rispettivamente computer autonomo e posto di lavoro intelligente), sono disponibili dalla metà di Maggio. I modelli a 8 bit, denominati con la stessa filosofia TS803 e TS800, sono anch'essi disponibili a partire dalla stessa data.

Televideo Systems Inc.
Indutrieweg 20G
1521 ND Wormerveer - Olanda
David Holdaway 075/287461

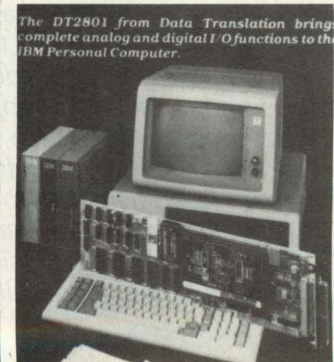
Compilatore Fortran per CP/M

Disponibile presso la Zelco di Milano un nuovo compilatore Fortran per CP/M denominato Pro-Fortran e prodotto dalla Prospero Software. Il compilatore viene fornito su dischetti a 8" singola faccia singola densità (formato IBM) oppure su minifloppy nei formati compatibili con le più diffuse marche di personal computer presenti sul mercato.

Zelco Srl
Via Vincenzo Monti, 21
20123 Milano
Tel. 02/803336

Due accordi e una scheda

La Eledra 3S, nota società di distribuzione di componenti e sistemi elettronici, rende noto di avere concluso due accordi commerciali. Il primo riguarda la rivendita delle stampanti Epson su tutto il territorio italiano. L'altra parte stipulante dell'accordo è la Segi che come molti sanno è l'importatore di queste stampanti. Il secondo accordo stipulato dalla Eledra riguarda il settore delle periferiche magnetiche ed è stato concluso con la Xebec Computers Corporation. Prodotto di punta di quest'ultima è un controllore per winchester a 5" "S1410" collegabile a tutte le periferiche magnetiche che usano il bus St506. Un terzo annuncio della Eledra riguarda l'acquisizione da parte di questa società di una scheda per l'acquisizione dei dati studiata per il PC IBM denominata DT2781. Ciò permetterebbe l'uso di questo personal in laboratorio come terminale di acquisizione dati e per applicazioni di controllo industriale.

Eledra 3S S.p.A.
V.le Elvezia, 18 Milano
Tel. 02/349751

Sorrida prego...

Un programma per Apple II e III recentemente introdotto permette di esaminare la dentatura di un paziente avvalendosi dell'uso dei citati personal computer. Si tratta di eseguire automaticamente alcuni esami dell'arco dentale in modo automatico, un po' come da un po' di tempo si fa con l'analisi della vista automatizzata. Il programma gestisce anche un plotter sul quale è possibile visualizzare l'arco dentale del cliente dopo averne tratteggiato il profilo su una tavoletta grafica. Inoltre il medesimo programma permette di tenere l'archivio della clientela organizzato in successione alfabetica. Questo dovrebbe agevolare il lavoro dei professionisti del settore odontotecnico. Inoltre, grazie allo schizzo dei punti cefalometrici del viso, sarà più difficile scambiare la dentiera di un paziente con quella di un altro.

Dr. Flavio Morosi
European society of medical software
Via Dalmazia 2 - Vigevano 27029 Pavia
Tel. 0381/42151

Novitissima

L'ultimissima novità della gamma dei pocket computer Casio si chiama PB-300. Sarà disponibile sul mercato italiano da Giugno o da Luglio. Questa unità è una versione migliorata del PB-100 (articolo su **Bit** n. 35 - Gennaio '83 a pag. 35) e include una stampantina su carta termica e

fino a 1568 passi di programma e fino a 222 registri di memoria senza espansione della stessa, cosa che non era possibile sul PB-100. Inoltre il distributore di questi prodotti rende noto che a partire da Maggio sono disponibili nei negozi le stampantine termiche, denominate FP-12, per il Casio PB-100.

Ditron S.p.A
V.le Certosa, 138
20156 Milano
Tel. 02/3085645

«Farò cose da meravigliare il mondo intero...

...con il GPS-4»

Sede: 20146 Milano, Via dei Gracchi 20, Tel. 02/4996 - Telex 332189; 40122 Bologna, Via del Porto 30, Tel. 051/522231; 00198 Roma, Via Paisiello 30 - Tel. 06/8448841 - Telex 610511; 10139 Torino, P.za Adriano 9, Tel. 011/443275/6-442321 - Telex 220181

DISTRIBUISCE

GENERAL PROCESSOR

ELABORATORI ITALIANI

Apple diventa PC-IBM

Una scheda prodotta negli Stati Uniti permette di aggiungere un microprocessore 8088 e 64 Kbyte di RAM a un Apple II, IIplus o IIe. Ciò dovrebbe permettere a chi sviluppa software di scrivere programmi per il PC IBM usando il proprio Apple (di cui ne sono stati venduti più di 750.000 esemplari nel mondo intero). La scheda che si chiama 88card viene fornita con l'MS-DOS e l'MBasic e costa al dettaglio 595 dollari negli Stati Uniti.

Personal Computer Products Inc.
16776 Bernardo Center Drive
San Diego, California 92128
Tel. 714/4858411

Teste pensanti e teste scriventi

La testa dell'ultima novità in fatto di ink jet dalla Siemens è una testa scrivente. Viene montata sulla PT 88, una stampante dotata appunto di tale tecnica di stampa, da poco giunta sul mercato. La stampante è versatile quanto ai tipi di carta (soffietto, rotolo e foglio singolo), veloce (150 caratteri al secondo), e silenziosa. La scrittura può avvenire in modo proporzionale con l'inserimento cioè di microspazi tra una parola e l'altra per ottimizzare l'aspetto estetico dello scritto. Possono essere stampati esponenti e discendenti. Si possono selezionare via software funzioni quali la tabulazione, la scrittura espansa, la spaziatura tra 10, 12 e 17 caratteri per pollice e la sottolineatura. Mediante una espansione di memoria è possibile lavorare con la stampante in modo grafico. Evidentemente senza una buona testa, questa volta pensante, tutto ciò non sarebbe stato possibile.

Per metterci dentro le mani

La Polar Electronics Ltd. ha presentato uno strumento che permette di localizzare i corto circuiti sulle schede a circuito stampato. Un rivelatore magnetico permette di seguire il percorso della corrente sulle tracce fino al guasto. Lo strumento è denominato Toneohm 580.

All data srl
Via delle Ande, 8
20051 Milano
Tel. 02/3087378

Emulatore per Z80 a 6 MHz in tempo reale

La SGS ha recentemente annunciato per il suo nuovo Computer di Sviluppo Compatto UX8-22, la disponibilità di un potente e completo emulatore per Z80.
Il software che correda la scheda UX8-EMZ80 e il pod contenente la CPU per l'emulazione, permette di eseguire operazioni di grande utilità. Tra queste la correzione dei programmi con il controllo contemporaneamente dei registri del microprocessore, della memoria e delle porte di I/O. È anche presente la possibilità del disassemblaggio del codice eseguibile. Con dell'hardware aggiuntivo (una scheda) si può eseguire la tracciatura in tempo reale del programma in esame visualizzando gli stati dei registri in un grafico che aiuta notevolmente a comprendere il funzionamento del programma. Anche la opzione di tracciatura, denominata RTZ80, opera in tempo reale fino a 6 MHz.

SGS-ATES Componenti Elettronici
Uffici commerciali di Milano
Tel. 02/4695651

Un canale da Londra

La Southdata LTD di Londra ha realizzato un data base manager per i micro-computer sotto CP/M con Microprocessore Z80.
Il pacchetto software si chiama Superfile e permette di ottimizzare l'uso dello spazio sul disco aumentando l'estensione dei file senza doverli riformare. Versioni a 16 bit per funzionare sotto il PC-DOS e l'MS-DOS, l'UNIX ecc., saranno disponibili per la seconda metà del 1983. La società inglese cerca distributori in Italia che siano anche in grado di tradurre i manuali.

Southdata LTD
10 Barley Mow Passage
London W4 4PH
Tel. 01 994 6477

Ci sono mille buone ragioni per comprare il personal computer HP 86.

Con tutte queste soluzioni non c'è più spazio per i problemi.

- Soluzioni per gestire tabelle elettroniche
- Soluzioni per lettere, rapporti e memorandum
- Soluzioni per gestire informazioni e banche dati
- Soluzioni per le rappresentazioni grafiche
- Soluzioni per la trasmissione dei dati.

E se queste non bastano, l'HP 86 ti offre anche soluzioni per ingegneria, meccanica, statistica, gestione, finanza ed altre ancora. Quest'ampia scelta di soluzioni e la possibilità di configurare modularmente l'HP 86, ti consentono di avere un sistema in grado di espandersi quando aumentano le tue necessità.

Piú tre ottime ragioni per comprarlo ora.

Fino al 15 luglio 1983 la Hewlett-Packard ti offre - al prezzo di uno - tre *package* fatti per accrescere la tua produttività: il "Personal Productivity Pac" include i *package* VisiCalc® Plus, File/80 e Graphics Presentations.

VisiCalc® Plus è un potente strumento di analisi che ti permette di creare fogli di lavoro e tabelle elettroniche. Cambi un dato, e tutta la tabella viene automaticamente aggiornata. È una risposta meravigliosamente semplice a tutti i "Che cosa succede se..." che incontri nella tua attività di analisi e di pianificazione.

File/80 ti consente di memorizzare e ritrovare rapidamente le tue informazioni, di aggiungere, modificare o cancellare dati e di gestire facilmente i tuoi archivi: il tutto elettronicamente, senza bisogno di schede e schedari.

Graphics Presentations ti consente di produrre, per mezzo di un *plotter*, diagrammi circolari o lineari, istogrammi e pagine di testo multicolori di qualità altamente professionale. E per le tue presentazioni puoi realizzare tutto questo anche direttamente su trasparenti per lavagna luminosa.

L'offerta del "Personal Productivity Pac" è indipendente dall'acquisto del computer.

I Rivenditori Autorizzati HP sono a tua disposizione per dimostrarti praticamente come l'HP 86 e il "Personal Productivity Pac" ti permettano di raggiungere nuovi livelli di produttività nel tuo lavoro.

Per ricevere ulteriori informazioni e il nome del Rivenditore più vicino, telefona allo 02-92369468 o spedisci il coupon alla Hewlett-Packard Italiana C.P. 10190 - 20100 Milano.

Desidero sapere tutto sull'HP86 e il "Personal Productivity Pac".

Nome e Cognome ____________________

Incarico ____________________

Società ____________________

Indirizzo ____________________ CAP __________

Città ____________________ Tel. __________

BIT/86 TR

VisiCalc® è un marchio registrato della VisiCorp.

Quando sono i risultati che contano

hp HEWLETT PACKARD

Bestiole d'altra specie

Fresca da Santa Palomba (protettrice di alcuni pesci?) giunge in redazione la notizia di un nuovo "piccolo" elaboratore IBM che viene cioè costruito. Ricordiamo che a S. Palomba è stato recentemente aperto il nuovo stabilimento della realcasa vicino a Roma. La nuova unità si chiama Sistema/36 ed è destinato a rappresentare una possibilità di crescita per gli utenti del Sistema/34, il più diffuso della IBM in Italia. Le caratteristiche del sistema sono una memoria centrale di 128 Kbyte espandibili a 512 Kbyte, memoria di massa su dischi winchester da 30 a 400 Mbyte e possibilità di collegare fino a 30 terminali locali e fino a 64 remoti. La CPU è composta da un insieme di processori (da tre a

sei a seconda dei modelli) che operano simultaneamente permettendo un tempo di elaborazione ridotto a minor possibilità di errore. Se diamo questa notizia una ragione c'è: vogliamo fare contare ai nostri lettori che il divario tra personal computer della fascia professionale e elaboratori di stazza superiore come viene naturale pensare parlando di questi sistemi, non è poi tanto grande.
Sulla strada delle insegne la IBM rende noto di aver messo a punto e pronto alla consegna un nuovo pannello di visualizzazione che si basa anziché sulla tradizionale tecnica del tubo a raggi catodici, su quello della scarica nei gas. Il video del pannello di visualizzazione è infatti composto di due lastre di vetro distanti meno di mezzo millimetro tra loro e la cui intercapedine è riempita di una miscela di neon e argon. Vi sono anche all'interno delle due lastre dei conduttori che fanno sì che il gas si illumini quando essi sono percorsi da corrente.

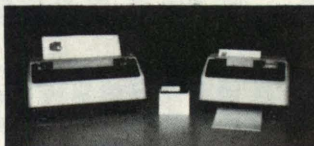
IBM Italia
20090 Segrate Milano
Tel. 02/75484550

Una nuova professione

Dal 13 al 15 Settembre si terrà al Centro Palexpo di Ginevra un duplice incontro. Si tratta della ripetizione di un analogo meeting tenuto per la prima volta lo scorso anno e che raccolse numerosi consensi. Il tema attorno al quale i due meeting ruotano è quello del "moto". Con questo si intende fornire un simposio internazionale che fornisca agli intervenuti una istantanea sullo stato dell'arte della ricerca privata nei vari paesi per quanto riguarda tutti i problemi connessi al movimento di parti qualsiasi. I rami dell'industria implicati sono molteplici e vanno dalla robotica alla pneumatica e oleodinamica alla scienza dei computer. Interverranno esperti di tutti i paesi europei e alcuni americani. Il nome delle due manifestazioni che si svolgeranno parallelamente al Palexpo sono PCI-Motor Con e Autofact. Il presidente del comitato promotore per l'Europa, Sig. Akram Yunas ha detto: "Mettendo assieme le varie tecnologie usate nel movimento di pezzi si può ipotizzare che questo dia luogo a una nuova professione, quella dell'ingegnere di movimento...".

Coordinatore europeo:
Mr. Gerd Zieroth
Incom Zieroth & Partner
Kleinreuther Weg 58
8500 Nurnberg 10
Germania Occidentale

Stampanti grafiche a colori

La Integral Data Systems Corp. produce una nuova serie di stampanti grafiche che non avranno mancato di attrarre l'attenzione di chi ha potuto amministrarle all'opera nelle ultime fiere del settore dei micro. La serie è denominata Prism 80/132 e può riprodurre grafici a colori con una risoluzione di 84 punti per pollice a una velocità di 200 cps. È disponibile il software

di gestione per interfacciare la stampante Prism ai più diffusi personal computer. Le porte di comunicazione disponibili sono la RS232 oppure la Centronics compatibile.

DDP Srl
L.go Migliara, 16 - 10143 Torino
Tel. 011/7497635

Polmoni di ferro

Dalla "operosa Brianza" un prodotto degno di considerazione. La Iper ha progettato uno spooler di stampa, più volgarmente un "polmone", denominato ISP-256, dotato di microprocessore e di una propria RAM da 256 Kbyte. Inserendo questo "polmone" tra un computer e la stampante, si può scaricare tutto il contenuto dei file che si desiderano stampare nella RAM del polmone in pochi secondi e riprende immediatamente il controllo dell'unità centrale. Il vantaggio è evidente specie se si considera il caso in cui si debbano stampare file molto lunghi (vi si possono memorizzare circa 60 pagine di una rivista come **Bit**). La stampante viene poi gestita dal microprocessore dello spooler che le invia i caratteri alla velocità appropriata. Lo ISP-256 è dotato di entrambe le interfacce più comuni, vale a dire la seriale RS 232 C e la parallela Centronics, sia all'ingresso dei dati che all'uscita.

Un secondo modello fornito dalla società costruttrice prevede due ingressi e due uscite. Ciascun ingresso può indirizzare una qualsiasi delle uscite. Quale sia l'affidabilità dell'hardware degli apparecchi proposti, non è ancora dato di sapere. Se si rileverà buona (esiste comunque una routine di autodiagnostica), visto che il programma che gestisce lo spooler può essere anche modificato, si può senza dubbio affermare che questi apparecchi rispondono a una esigenza professionale molto viva.

Iper
Via Giovanni XXIII, 19
22070 Limido Comasco (CO)
Tel. 031/938311

Raccomandata in redazione

La Cattaneo Systems di Genova rende noto che a partire dalla fine dello scorso anno non è più importatrice per l'Italia della Intertec e quindi dei suoi prodotti Superbrain e Compustar. È invece attivamente impegnata nella distribuzione della Sord Computer Corporation di Tokio, della Wicat Systems Inc. di Orem (Utah) Stati Uniti, General Electrin Company di Waynesboro (Virginia) Stati Uniti, e BMC International di Osaka in Giappone.
Con questo speriamo di aver fatto il nostro dovere avvertendo i lettori e evitando complicazioni alla suddetta azienda nei rapporti con gli attuali importatori della Intertec.

Un fac-simile versatile

La 3M ha presentato recentemente un tele-copiatore fac-simile denominato 9145 che funziona un po' come un telex e un po' sul concetto della posta elettronica. È possibile interrogare mediante il 9145 diversi terminali remoti vedendo se questi hanno messaggi da inviare (anche in assenza dell'operatore mediante una speciale funzione automatica). Nel caso positivo questi vengono ricevuti e il 9145 mette automaticamente su ciascun messaggio un identificativo del mittente, la data e l'ora di ricezione. Il telecopiatore si sintonizza inoltre automaticamente sulla velocità di trasmissione del terminale remoto. Può riprodurre copie con risolvenza di 7.7x7.7 linee per millimetro e utilizzare scritture cifrate in trasmissione. Ultima caratteristica di questa versatile periferica è che può fungere da terminale stampante di un video terminale o di un computer.

3M Italia S.p.A.
20090 Milano S. Felice - Segrate
Tel. 02/75452595

Microelaboratori: Uno studio delle prospettive fino alle soglie del 2000

di **A. Cavalcoli**

Parte seconda

Dopo la panoramica generale vista nella prima parte, addentriamoci ora in aspetti che sempre più coinvolgono nuovi scenari della vita quotidiana e gli impatti sul mercato del lavoro.
Per una maggior diffusione dei microelaboratori, sono sottolineati i seguenti fattori favorevoli:

- la logica del consumatore, che vede nel microelaboratore un bene durevole;
- la possibilità di risolvere piccoli e grandi problemi personali;
- la rapidità con cui i giovani si impadroniscono del loro uso;

Sfavorevoli sarebbero invece i seguenti:

- inesistenza di una cultura tecnica specifica;
- concorrenza di altri "oggetti elettronici" di consumo (videoregistratori, ecc...). Su ciò non siamo molto d'accordo, trattandosi di cose ben distinte.
- Futura concorrenza di altri prodotti disponibili tramite reti telematiche (diverso e nuovo uso del telefono).

A questo punto si entra in un aspetto molto interessante, in collegamento con il futuro habitat urbano e l'evoluzione dell'ambiente familiare, caratterizzato anche da una sempre maggior presenza femminile nel mondo del lavoro.
Si parte da tre scenari:

- rafforzamento dell'habitat attuale, cioè la casa, la vita familiare come è oggi vissuta.
- apertura dell'habitat: creazione di significative comunicazioni tra il mondo della casa e quello del lavoro, da cui un arricchimento.
- impoverimento delle funzioni in seno all'habitat, ad esempio uso della casa solo per ritornarci alla sera, per dormire.

Seguono tre distinte evoluzioni. Vediamole separatamente, ciascuna in funzione di vari fattori.

1) Rafforzamento dell'attuale

a) Fattore *"estensione del lavoro femminile"*. Si prevede una diffusione dei microelaboratori a livello piccola gestione domestica. Ancora, il microelaboratore potrà essere alla base di forme più o meno automatiche di gestione del "confort" casalingo (riscaldamento, luci, porte, ecc.).
b) Fattore *"estensione del tempo libero"*. Ecco apparirne le tipiche implicazioni, quali i giochi.
Non si escludano attività di autoistruzione, autoinformazione medica di base, acquisti via rete, con servizi rionali o cittadini. Si prevede una diffusione modica.
c) Fattore *"incremento di richiesta in materia di sicurezza"*. Sistemi di sicurezza sia dedicati che collegati a reti (forte diffusione).
d) Fattore *"Evoluzione della nozione di benessere sociale verso un controllo esterno"*.
Applicazioni multiple, sanità, dietetica. Arti divinatorie, test psicologici tramite microelaboratore. Diffusione massiccia di microelaboratori di tipo definito: "polygadget".

2) Arricchimento delle funzioni esercitate in seno all'habitat

a) Fattore "estensione del lavoro femminile". Applicazioni di ogni genere, con confusione tra domestico e professionale.
b) Fattore "estensione del tempo libero". Grande spazio ai giochi. Acquisti via video. Educazione assistita dal computer. Stabilirsi molto forte di attività d'autoproduzione ed autonome. Spiccato sviluppo delle reti, in funzione delle quali si avrà il boom della diffusione.
c) Fattore "sicurezza". Diffusione polifunzionale del microelaboratore.
d) Fattore "evoluzione della nozione di benessere sociale verso un controllo esteso". Massiccia diffusione dei microelaboratori, ma articolazione domestica/professionale ancora da definire.

3) Impoverimento dell'attuale habitat

a) Fattore "estensione del lavoro femminile".
Il microelaboratore è visto come oggetto di razionalizzazione: debole diffusione.
b) Fattore "tempo libero". Nessuna diffusione dei microelaboratori in sede domestica, ma negli spazi collettivi.
c) Fattore "sicurezza". Bassa diffusione di microelaboratori "cani da guardia".
d) Fattore "benessere". Forte diffusione dei microelaboratori nello spazio collettivo.

Alla fine di questa carrellata sui futuribili discende chiaramente che una grande diffusione dei microelaboratori è strettamente legata ad un armonico sviluppo del nostro habitat sarà più ricco ed articolato e offrirà maggiori momenti applicativi a questi oggetti di razionalizzazione, di divertimento, di cultura, di lavoro.
E a questo proposito conviene passare all'ultima parte, in cui si pone la cruciale domanda: *"saranno creati nuovi posti di lavoro?"*.

Microelaboratori e mercato del lavoro

Dopo aver analizzato il mercato, si passa ad una valutazione delle attività indotte dalla microinformatica fino all'orizzonte del 1995.
Gli esperti, i costruttori, i distributori intervistati, in pratica rifiutano di fare previsioni per un periodo superiore ai 5 anni, a causa della forte dinamica tecnologica, che può originare situazioni, del tutto nuove. Del resto si legge che la tendenza sarebbe *"non tanto quella di creare nuovi posti di lavoro, quanto di valorizzare presso i potenziali utenti le economie in termini di manodopera"*.
Questa frase è emblematica e contiene problematiche sociali non indifferenti. Il concetto è ripreso più avanti, dove parlando di produzione di microelaboratori, si sottolinea la tendenza verso una automatizzazione spinta.
Un nuovo spunto proviene poi dalla futura attività di manutenzione: manutenzione tradizionale o telemanutenzione? Alla fine si ha un clima di grande incertezza e diversamente non potrebbe essere anche perchè

Microelaboratori: uno studio delle prospettive fino alle soglie del 2000

occorrerebbe effettuare, come dicono gli estensori dello studio, una analisi congiunta dell'evoluzione di ben quattro variabili: costo dell'hardware, volume di vendite; produttività; fatturato per addetto.

a) *Previsione del mercato e creazione di posti di lavoro.*
Basandosi su dati raccolti da una società di consulenza, la SEMA Consultronique, si ha, con riferimento ai cosiddetti prodotti di bassa gamma, valido per la CEE (figura 1)

	1979	1981	1983	1985
Valore (milioni di $)	27	55	110	285
Numero (migliaia)	28	80	240	800
Prezzo medio (in $)	964	687	458	356

Figura 1

Si può notare il costo base estremamente basso per ogni sistema: ci si riferisce a prodotti indicati nello studio come: "Ordinateur domestique".
Dal punto di vista della produzione, i sistemi sarebbero realizzati:
- 15% da costruttori europei nella CEE;
- 15% da costruttori non europei nella CEE;
- 70% importati nella CEE.

Sulla base di considerazioni più o meno accettabili, esce il numero di 1735 nuovi posti di lavoro creati da qui al 1985.
Passando ai sistemi definiti di gamma elevata (costo medio attuale 5000 $) vale quanto segue, sempre sulla base della ricerca Consultronique (figura 2)

	1979	1981	1983	1985
Valore (milioni di $)	300	680	1450	2900
Numero (migliaia)	60	120	300	600
Prezzo medio (in $)	5000	5600	4800	4800

Figura 2

	1978	1979	1980
Apple Computer	10	60	165,2
Tandy	65	150	210
Commodore	—	64	99

Figura 3

Queste cifre non sono molto chiare, non essendo precisato se si tratta di sistemi prodotti o venduti.
Altrove si prevede, per i sistemi di gamma alta, un parco di 1.600.000 sistemi nel 1985.
Alla fine, sempre sulla base di considerazioni variamente accettabili, si parla di 49.545 nuovi posti di lavoro da qui al 1985 come totale del mercato gamma alta e bassa, cui si aggiungono 4220 altri posti associati al settore Home Video-text Terminal.

Gli esperti interrogati dagli estensori dello studio prevedono un volume di vendita di 6 milioni di unità per il 1995 nella CEE, con un prezzo medio per sistema di 250 $.
Alla fine si è molto sconcertati da queste cifre, oltre che da altre valutazioni numeriche che qui non riportiamo.
Anche gli autori dello studio sono perplessi: *"tenuto conto delle numerose incertezze che pesano a lungo termine sulla validità dei rapporti di creazione di posti di lavoro".*

b) *La produzione di microelaboratori e componenti.*
Tre sono le ditte che attualmente dominano il mercato: Tandy Radio Shack, Apple Computer e Commodore.
In figura 3 i volumi d'affari (in milioni di $)
Si fa notare che tra queste, ad esempio la Commodore dispone della sua propria materia prima, possedendo la società che produce il microprocessore 6502.
Se quindi da un lato costruttori di microelaboratori posseggono od ampliano la loro partecipazione in industrie di semiconduttori, dall'altro alcuni di questi tendono ad entrare nel settore microelaborari (I.T.T., T.I., Zilog).
Dal punto di vista della creazione di posti di lavoro questo settore appare perciò assai dinamico e promettente, anche se impone un salto culturale degli addetti, cosa di cui si parlerà più avanti.
Carta vincente è il software, non tanto quello applicativo, quanto quello di "sistema", ma si torna così di nuovo al discorso delle future competenze.

c) *Distribuzione e manutenzione*
La Tandy Corporation appare essere il leader nella distribuzione:
6000 negozi in USA nel 1979; 8000, sempre in USA, nel 1980 e 2000 in Europa nel 1980.
Gli 8000 punti di vendita sono in pratica un valore di stabilizzazione (un negozio ogni 25.000 abitanti).
E per la CEE? Con una estrapolazione sui dati precedenti e nell'ipotesi di 330 milioni di abitanti nel 1995, si arriva a 13.200 negozi di elettronica "grand public" nel 1995. Ora, se vediamo almeno due addetti per negozio, abbiamo attorno a 26.400 nuovi posti di lavoro nella rete di vendita.
Sulla base delle attuali tendenze si può poi ipotizzare:
- la creazione di centri di distribuzione/assistenza (con questo verrebbero eliminati i distributori cosiddetti indipendenti) da parte dei costruttori;
- un nuovo impulso nella vendita per corrispondenza;
- la presenza di microelaboratori anche nei grandi magazzini.

d) *Le reti di microelaboratori.*
Sono già operative molte reti di microelaboratori: MICRODIAL, MICRONET, THE SOURCE, PC-NET.
È un fenomeno ancora tipicamente americano, che potrebbe essere importato nella CEE, da cui:
- rischio di una dipendenza culturale (dicono gli autori: *"diffusione di informazioni a carattere politico in un involucro neutro e scientifico")*
- rischio economico, dato il carattere più "avanzato" e "smaliziato" delle industrie culturali americane.

In generale, una forte concentrazione di servizi accessibili tramite reti internazionali può destabilizzare alcuni servizi tradizionali (poste, giornali, piccole attività commerciali, ecc.) da cui impatto negativo sul fronte occupazionale. Secondo gli esperti interrogati, tra il 1990 ed il 2010 circa il 50% delle abitazioni saranno attrezzate con terminali per accedere a banche di dati.
Ostacoli alla diffusione delle reti sono: il costo, l'esigenza di un minimo di specializzazione, la concorrenza di altri prodotti (videodischi, dicono nello studio), i problemi di sicurezza dei dati, riservatezza, ecc.
Alla fine, *"molto aleatorie"* sono le previsioni di creazione di posti di lavoro, associate alle reti.

e) *Formazione*
In tanta nebbia, incertezza, aleatorietà, finalmente una vivida luce. *"Il settore che a breve termine può rivelarsi come il maggiore creatore di posti di lavoro è quello della formazione all'informatica".*
Spazio è dato anche alla educazione assistita dell'elaboratore (CAL, CAI), alle operazioni editoriali connesse, alla vendita di software educativo, ma di un certo tipo, più serio e curato dell'attuale.
Essendoci di mezzo la scuola, la pubblica istruzione, non solo quelle a carattere privato, correttamente si afferma che la maggioranza di questi *posti non* può che essere creata da una volontà politica.

La nostra conclusione

Fatti salvi alcuni dati precisi, molte delle valutazioni dello

Il mercato pubblico

Questo mercato è stato tra i primi ad essere aggredito ed è anche stato il primo a configurarsi come banco di prova per i primi prodotti, verso proposte più ottimizzate e centrate dei costruttori.
Stranamente, gli estensori dello studio giudicano molto debole la penetrazione attuale in questo mercato, portando due cause prime:

- pochi prodotti disponibili (sia in termini di sistema che di software);
- scarsa familiarizzazione del pubblico verso questa nuova tecnologia

Giudichiamo strano questo pregiudizio, soprattutto con riferimento alla prima delle due cause, restando d'accordo sulla seconda, da cui un ulteriore richiamo ad una formazione più agile, più accessibile.
Più interessante e potenzialmente notevole il mercato dei microelaboratori nelle collettività locali.
Si parla in questo caso di comuni, anzi di piccoli comuni, dai 1500 ai 5000 abitanti.
Una premessa: si dice, nello studio, che se da un lato i microelaboratori possono avere un ambito applicativo comunale ideale, dall'altro problemi politici e di gestione del potere rappresentano ostacoli insormontabili.
Tra gli altri ostacoli, i cronici problemi finanziari, e l'assenza di una prospettiva a lungo termine, con questo intendendo tutto l'insieme di incertezze legate agli sviluppi politici ed economici che, alla fine, vincolano e pilotano l'essere dei comuni stessi.
Gli utilizzi individuati sono soprattutto gestionali e legati ad una visione di accresciuta efficienza e di migliorato servizio.

studio sono basate su ipotesi talora difficili da condividere, benchè ragionevoli.

Non è una critica in assoluto, ma una riprova della quasi impossibilità, spesso sottolineata in modo onesto dagli autori, di dar vita a precise linee di sviluppo su tempi lunghi, in un settore così dinamico e sfuggente, che è poi la sua ricchezza se non il suo fascino.

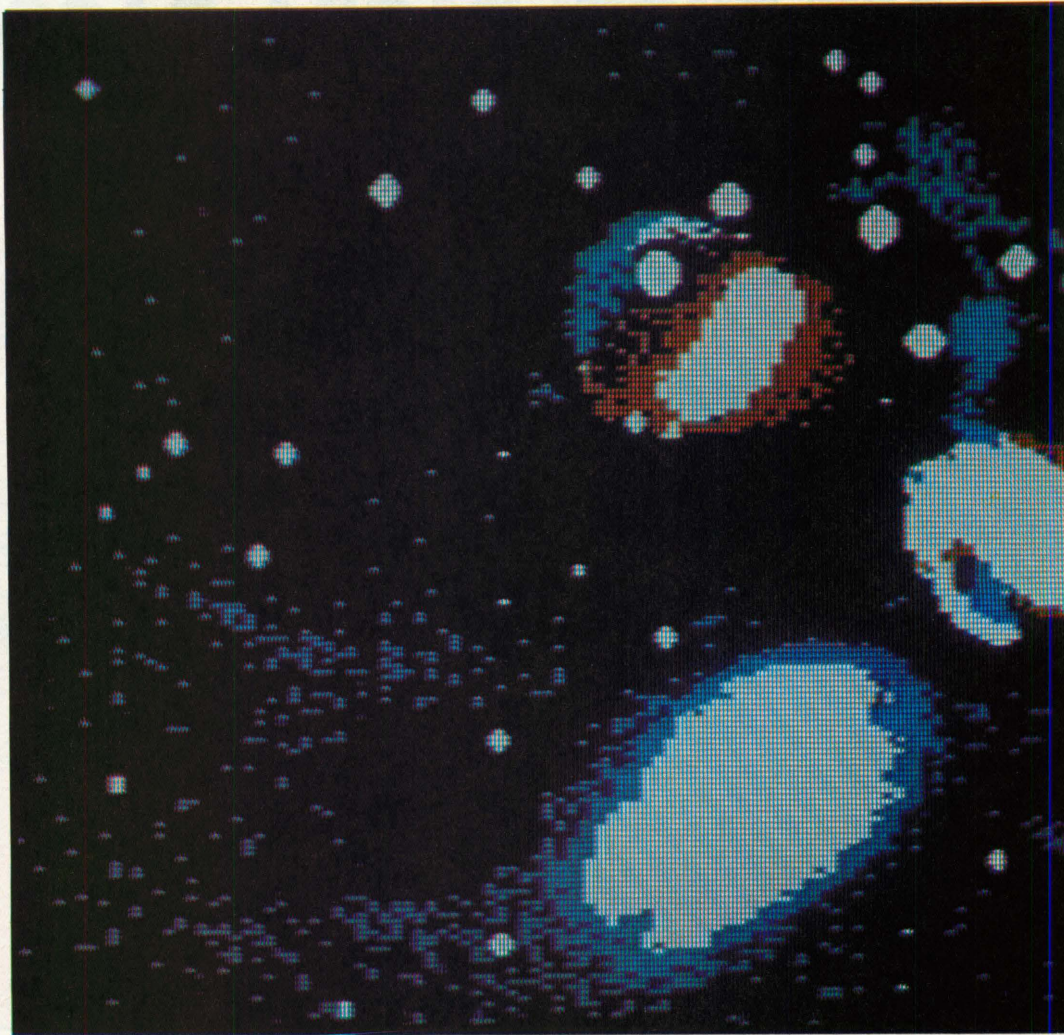
In definitiva: se c'è un settore che corre è può dare soddisfazione, è questo. Spesso le regole del gioco sono nuove, come nuovi gli ambienti applicativi con un'utenza mai prima d'ora in contatto con qualcosa di elettronico. L'unica previsione sicura è quella globale, di un enorme sviluppo.

Infine teniamo a precisare, per rispetto ai ricercatori, che se lo studio presentato può apparire per alcuni versi carente o superficiale, l'impressione può esser frutto dei nostri commenti e delle nostre, inevitabili, omissioni.

Esso ha invece parecchi meriti, non certo ultimo quello di avere aperto un dibattito su un fenomeno così prorompente, anche se dai contorni alquanto sfuggenti.

■

Intervista ad
Adriano Abbado

Ad Adriano Abbado, 25 anni, proveniente da una famiglia milanese di musicisti (il noto direttore dell'orchestra della Scala è suo zio) collaboratore tra l'altro della rivista Jackson *"Strumenti musicali"* chiedo innanzitutto di raccontare per quali vie e nutrite di quali suggestioni sia prevenuto alla computer art, anzi alla "personal" computer art.
Risposta Da tempo sono attratto dalla comunicazione non verbale, audiovisiva, e dalla fotografia sono andato presto scivolando nei meandri della sperimentazione, affascinato dalle interazioni della luce e dei colori ottenibili per sovrapposizione di diapositive o utilizzan-

LA GEOMETRIA DELL'INEFFABILE

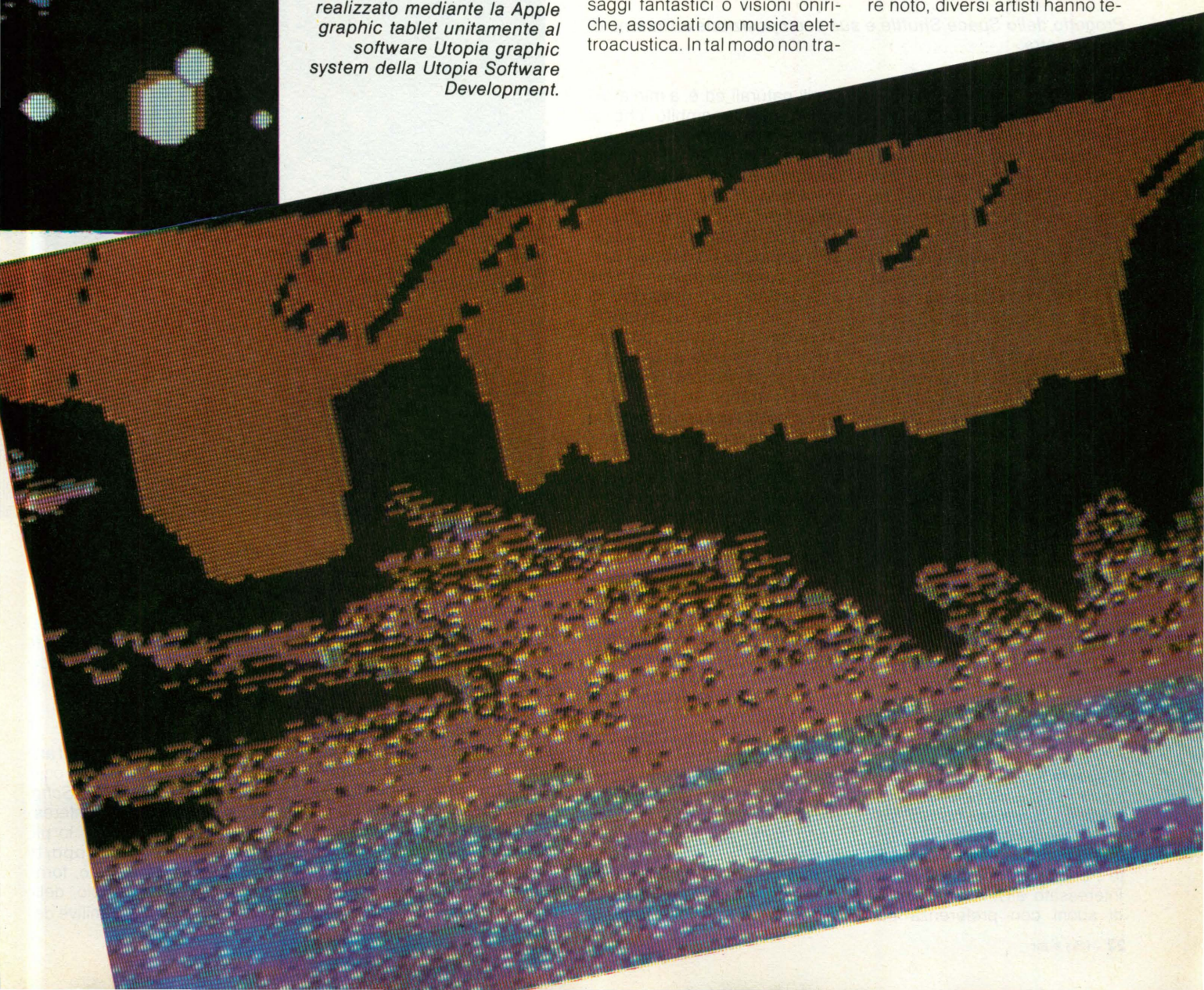
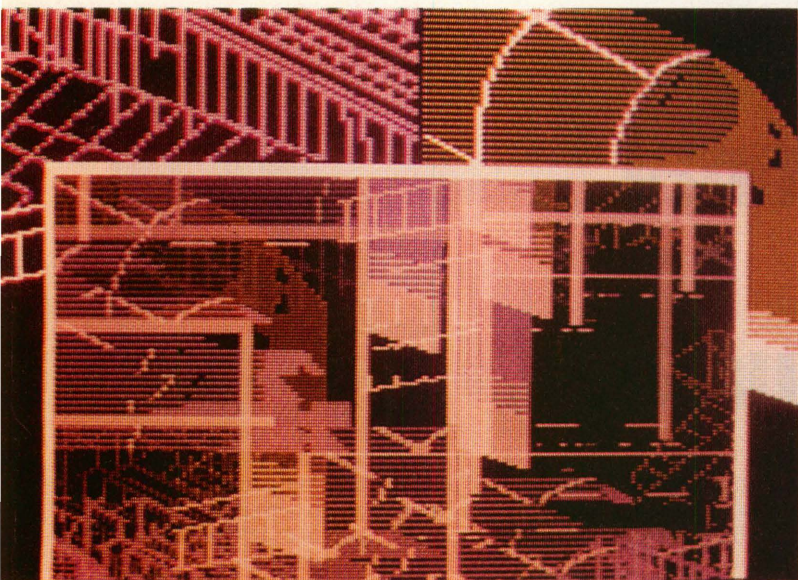
Inaugurando questo servizio, primo speriamo, d'una serie dedicata ad artisti che si servono del personal computer per alimentare la propria ispirazione più o meno avanguardosa, ci viene in mente chi sa perché la musica degli inizi del rock'n roll, il rock "classico", insomma.
Conquistò i giovani di colpo perché, a parere di chi scrive, subito seppe palesare due anime: quella dionisiaca esplodeva nella rabbiosa allegria di "Rock around the clock" ma c'era pure quella apollinea, un poco sdolcinata, a suscitare brividi sottili con le note struggenti del capolavoro dei Platters, "Only You". Se è vera questa teoria, la coerenza monotona d'uno stile, anche se raffinato, non basta a decretare un successo indiscusso e lo provarono, ci sembra, i Beatles con la loro capacità di giocare bene su entrambe le corde del lirismo e della rabbia.
Che c'entra questo col computer? diranno tutti. Non troppo sembra, dal momento che si tratta di strumento che conserva un prevalente carattere freddo, tipico degli oggetti partoriti dalla testa di gens mathematica. Imparziale e neutrale - "calcolatore", per l'appunto - il personal computer ha però saputo nel frattempo mutuare un carattere più libero e giocondo da altri settori. Quello dei videogame, ad esempio, e che questi si collochino ai confini tra frivolezza ed arte con *l'A maiuscola* è ormai giocoforza ammettere anche dai più restii dopo la fuoriuscita del film *"Tron"*, comunque se ne vogliano giudicare i risultati estetici, senz'altro di livello rispettabile. L'arte computerizzata, è stato detto da qualcuno di noi più autorevole, nasce d'altronde proprio da questo contrasto tra la naturale frigidità del docile ma versatile strumento e la calda temperie dell'artista. Gelato ricoperto di cioccolato bollente, se ci si consente questa immagine consumistico-barocca mirante a fornire una possibile chiave di fruizione di lavori come questo. Di fronte ai quali l'invito è di non lasciarsi andare agli opposti lapidarismi del *"non mi piace"* o del *"mamma che bello!"*.
Oltretutto, nel caso specifico, queste foto perdono moltissimo dalla mancanza della musica. con la quale formano binomio pressoché inscindibile. Occorre una riflessione, uno sforzo critico che, peraltro, non possono sfociare in un giudizio trinciato con l'accetta di precise parole. Questo è un fatto di sempre, con le arti visivo-uditive e la moderna fisiologia associata agli studi informatici di punta ce ne dà ragione: ormai lo sanno anche i ragazzini che il nostro encefalo è diviso in due emisferi e che su una metà risiede il raziocinio mentre nell'altra è la sfera dell'immaginazione creativa. In qualche modo si può forse dire che il computer è come se possedesse un semiencefalo più un pezzetto dell'altro e l'opera dell'artista consiste nell'aggiungervi ciò che manca. Nasce così quella che ci vien voglia di chiamare *geometria dell'ineffabile* e, malgrado la povertà pixel-espressiva dei personal più spartani, l'inesprimibile riesce nei casi migliori a travalicare i limiti di banda passante espressi dalla (ristretta?) Teoria dell'Informazione. E quanto all'ineffabile a noi sembra di cogliere - almeno negli esiti più alti di questa arte nuova nei mezzi ma antica nell'umanità - tanto l'apollinea dolcezza dell'eleganza linear-melodiosa quanto l'indicibile angoscia di questi tempi d'incalzante tecnologia. Esaltanti ma anche inquieti. Ci può essere qualcosa di meglio del personal computer per cogliere l'urlo e il furore?

G.G.

Le vie del personal all'arte audiovisuale

Galassie. Le immagini sono tratte dal film "Viaggio".

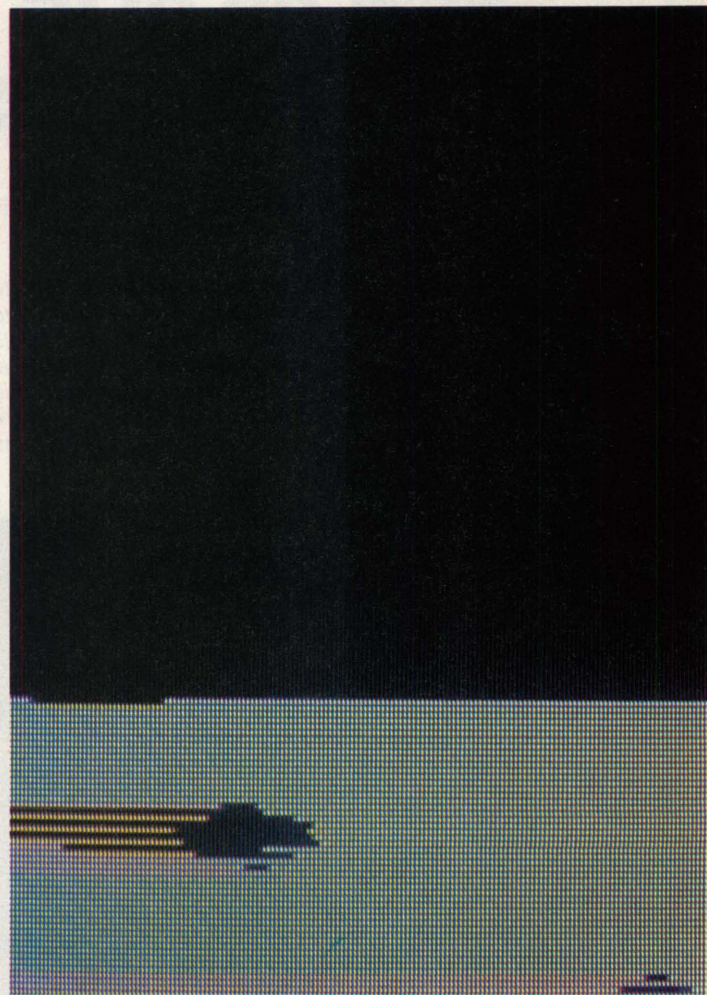
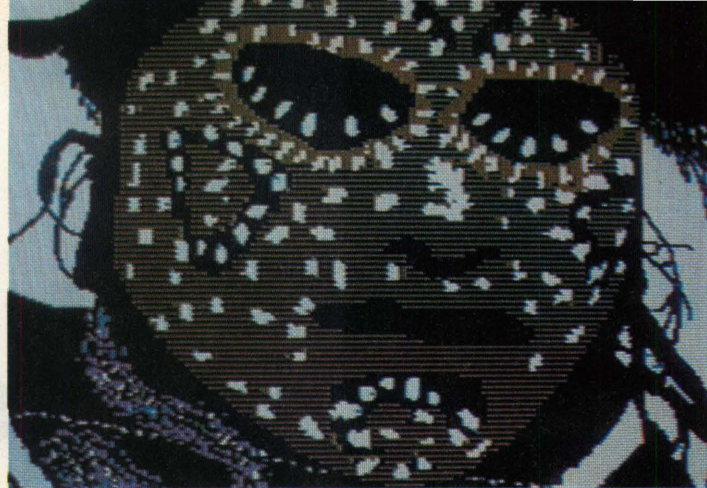
Paesaggio terrestre. Questo disegno, come altri, è stato realizzato mediante la Apple graphic tablet unitamente al software Utopia graphic system della Utopia Software Development.

do pittura, tempera ed altri materiali su pellicola, fino a realizzare pellicole super 8 in cui immagini distorte o variamente modificate suggeriscono paesaggi fantastici o visioni oniriche, associati con musica elettroacustica. In tal modo non tradivo certo la mia vocazione musicale: a quell'epoca già studiavo composizione musicale elettronica al Conservatorio di Milano e più passava il tempo più mi appassionavo ai problemi del rapporto tra musica ed immagine. Un rapporto nato con il cinema, anche se questo ha quasi sempre relegato in secondo piano la musica, ancella della vicenda e in genere limitata alla funzione di commento emotivo. Come dovrebbe essere noto, diversi artisti hanno te-

Progetto dello Space Shuttle e sua scomposizione all'interno di una finestra.

orizzato o posto in pratica l'idea d'una più intima fusione audiovisuale messa in atto con i mezzi che la tecnologia pone a disposizione e, anche se il loro sperimentalismo sovente ha avuto discutibili caratteri intellettualoidi, per lo più hanno contribuito positivamente ad aprire nuovi orizzonti. Personalmente ritengo che sperimentare sia un ambito mentale naturale per un artista del nostro tempo, se non altro per la continua curiosità di percorrere vie nuove, tuttavia l'obiettivo non deve essere un prodotto d'élite. Tornerò più avanti su tale concetto, adesso termino di rispondere alla tua domanda affermando che lo sbocco logico di quel desiderio di sposare più intimamente musica elettronica ed immagini, è infine stata l'arte attuata mediante il computer.

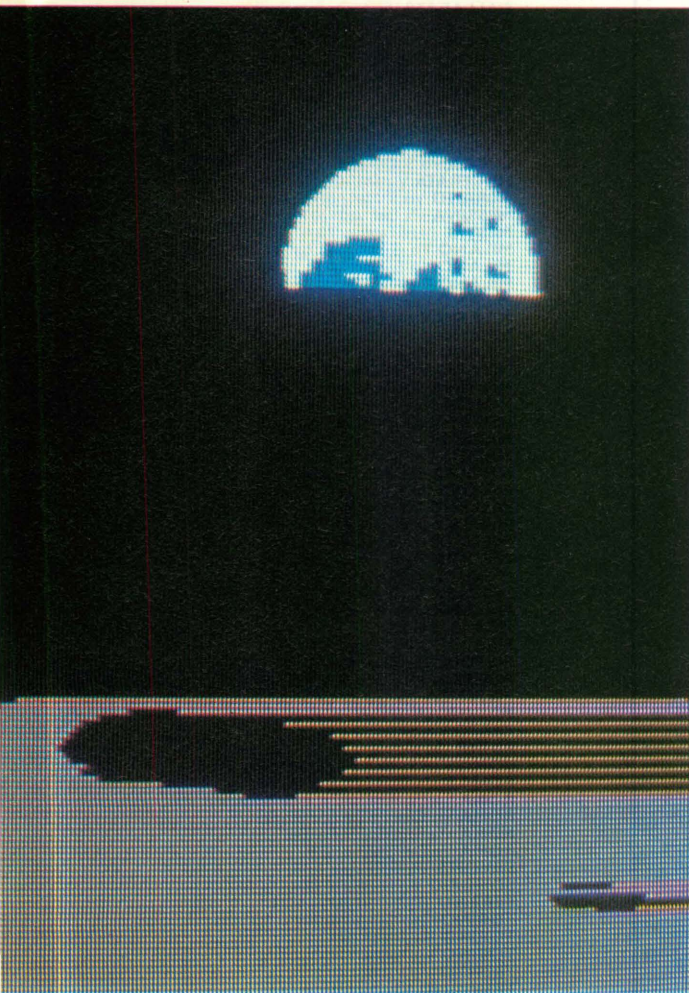
Domanda *A proposito di musica sperimentale con l'elaboratore, non ti sembra che troppo spesso i risultati siano stati deludenti rispetto alle promesse? In termini di calore, soprattutto...*

Risposta Si tratta di un universo molto vasto e complesso e nel termine "computer music" sono racchiuse potenzialità enormi ed estremamente ramificate. Per quanto mi riguarda, sono interessato all'analisi e sintesi di suoni con preferenza per quelli naturali ed è, a mio avviso, in questo ambito che si stanno ottenendo i risultati più validi e che, ne sono convinto, dovranno prima o poi "sfondare" anche a livello di diffusione nel gusto e nell'immaginario collettivo. D'altronde già adesso il computer ed anche il personal computer trovano mille applicazioni: luci, mixaggio di suoni e anche scenografie "elettroniche", il tutto, potremmo dire, in modo spesso trasparente al fruitore. Il fatto è che ci vorrebbe un teatro pensato e costruito espressamente per la comunicazione elettronica audiovisiva, secondo me dotando ogni spettatore del suo monitor a colori e di cuffia stereo non a contatto della pelle e magari includendo sistemi di trasmissione di profumi e sensazioni tattili. Quanto alla computer music anch'io confesso che solo raramente ho udito cose esaltanti, il che dipende anche da errori d'ambientazione. Ad esempio le enormi casse sonore ben in vista al posto dell'esecutore o il volerlo in qualche modo resuscitare con le esecuzioni "live", in tempo reale, non giovano certo a creare il calore delle esecuzioni di musica classica o di un concerto rock. A questo punto li preferisco.

A mio parere la strada giusta non è certo quella dell'imitazione di forme artistiche tradizionali, semmai occorre ispirarsi ad esse in modo non pedestre, rifacendosi ad esempio all'opera o, meglio, ai balletti. Per questo sono un fautore dello schermo visivo. Altrimenti l'attenzione dello spettatore si perde. Questo, ripeto, non vuol dire affatto assegnare alla musica un ruolo subordinato. Inoltre viaggiando in paesi lontani, ho provato l'emozione della scoperta di antichissime, primordiali radici di questo discorso. Ho potuto accostarmi a manifestazioni ancestrali di certi popoli primitivi, come quelli della Nuova Guinea o del Mato Grosso, presso i quali, nei riti e nelle danze, l'espressione audiovisiva è un fatto culturale tutt'ora vivo.

Domanda *Dunque anche l'arte del futuro ha un cuore antico. Ma tra i moderni a chi principalmente ti ispiri?*

Risposta Posso riferirmi ad artisti famosi della cultura occidentale, Klee, Kandiskij, Schoenberg, che sono stati interessati in vario modo (per lo più teorizzando) al rapporto musica-immagine. Però, torno ad insistere, l'esempio delle culture cosiddette primitive de-

uest'immagine si ispira ai sistemi di pittura del viso e del corpo uso presso diverse culture primitive, in questo caso brasiliana.

a Terra vista dalla Luna.

ve incoraggiare: pur nell'irrepetibilità del loro mondo nel nostro debbono stimolarci a costruire qualcosa di non intellettuale, bensì spettacolare, divertente e percettivamente stimolante.
Per quanto si riferisce ai nostri giorni molto valido per me è il libro "*Digital harmony*" di John Whitney (*delle BYTE editions della McGraw-Hill* NdR) in cui ha esposto ed esemplificato in modo stupendo le sue ricerche su ciò che egli chiama "video-music", stabilendo il principio di un codice di corrispondenza tra suono e luce. Non dimentico però il canadese Norman McLaren.

Domanda *Quali sono secondo te i vantaggi offerti dal computer per la pratica attuazione di simili principi?*
Risposta ovviamente innumerevoli, non ultimo quello dell'animazione come già l'esempio stimolante del film "Tron" rivela. In generale il pregio del computer consiste nella possibilità di pilotare, con una serie di istruzioni, diverse periferiche, tipicamente nel nostro caso altoparlanti e monitor. Quindi con un singolo programma, addirittura con la stessa istruzione, si può agire in modo coordinato sui cannoni elettronici del monitor e sui coni degli altoparlanti per realizzare quella corrispondenza tra suono e luce che è materia del moderno ricercare dell'artista audiovisuale.
Domanda *Ma il personal, per la musica, non ha ancora le qualità necessarie (velocità, adeguata risoluzione del video) per gli obiettivi più ambiziosi. Qual è stata la tua esperienza al riguardo?*
Risposta È fuori dubbio che il personal non può, oggi, competere in questo campo con i grandi elaboratori o con le attrezzature dedicate. Ma le sue possibilità non sono, a mio parere, limitate al punto da impedire l'attuazione di una ricerca e di un'attività espressiva che lo valorizzino come efficace mezzo di comunicazione.
Da circa un anno e mezzo lavoro con un personal, impiegandolo in modo ibrido, cioè unitamente ad altri strumenti. In questo non penso vi sia nulla di "impuro", anzi - pur rimanendo aperto a tutti gli eventuali suggerimenti e scambi di idee - ritengo che ad esempio l'impiego delle videocassette sia un fatto importante, nella prospettiva ambiziosa ma in cui credo fermamente, di un progressivo allargamento del mercato dei fruitori, dato che l'impianto casalingo di audiovisione sta diventando il culmine di un processo che li vede come elettrodomestici culturali. È stimolante proporre un diverso tipo di espressione e di linguaggio attraverso mezzi di diffusione culturali non esclusivi, in continua evoluzione. Non dimentichiamo che si sente sempre più spesso parlare di televisione stereofonica e ad alta definizione.
Quello da me realizzato è un film elettronico su videocassetta, che, come dovrebbe apparire chiaro da tutte le cose che ho detto, è un incrocio tra un audiovisivo e un cortometraggio, con la massima preoccupazione del bilanciamento delle informazioni musicali con le immagini. Come accennavo prima, di solito si ha un grande squilibrio a favore delle immagini. L'attenzione è su queste. Ho cercato di correggere il fenomeno rallentando il ritmo delle immagini e, viceversa, concentrando notevolmente i suoni. È chiaro che questo è solo il primo passo in questa direzione.

Autoritratto dell'Autore.

Il film "Viaggio" è una composizione per monitor e cuffia stereofonica di origine interamente elettronica, la cui musica ha origine da strumenti analogici, mentre le immagini sono derivate da un sistema digitale.
L'intera composizione è suddivisa in sei blocchi aventi una durata rispettivamente di 1, 1, 2, 3, 5, 8 minuti. Ogni numero è la somma dei due precedenti, ovvero si tratta dei primi sei della

Le vie del personal all'arte audiovisuale

celebre progressione di Fibonacci, matematico toscano del XIII secolo, che scoprì una relazione analitica di questo tipo in alcuni fenomeni naturali. Ad esempio la si riscontra in tutte le manifestazioni in cui è in gioco la spirale logaritmica, come nelle galassie, nelle chiocciole e così via. Mentre il primo minuto è dedicato ai titoli di testa, gli altri cinque blocchi costituiscono la composizione vera e propria. Una breve pausa-buio divide un blocco dal successivo. La musica ha, come ho detto, origine analogica, ossia è stata ottenuta con sintetizzatori e strumenti simili, ma sono presenti anche voce, gong, sibilo di reattori a turbina e chitarra elettrica. È chiaro comunque che tutti i suoni elettronici avrebbero potuto essere prodotti con un sistema digitale, anche quello di un personal. Diciamo che la scelta dell'analogico è dipesa dalla disponibilità delle apparecchiature. Alla fase di sintesi e registrazione sonora è seguita quella dell'elaborazione. Per tutte le operazioni musicali mi sono avvalso dell'aiuto di Riccardo Sinigaglia.
La colonna visiva è costituita da una sequenza di 53 immagini, rappresentate da disegni generati con la tavoletta grafica di un personal Apple II plus. Successivamente è stato creato, insieme a Michele Bohm, un programma per il computer avente la funzione di inventare delle dissolvenze, in totale dodici (naturalmente se ne potrebbero fare molte di più).
Terminata la fase del raccoglimento dei materiali, si è proceduto al montaggio su videotape, effettuato presso il Centro Televisivo Universitario. In primo luogo è stato fatto il riversamento della musica, che è così servita da riferimento per i tempi delle immagini. Quindi è stata montata la collezione di dissolvenze precedentemente ottenuta. La composizione, della durata di venti minuti, ha per soggetto un ipotetico viaggio nello spazio-tempo, attraversando iconografie stellari e graffiti preistorici.
Ora ho realizzato una versione arricchita, di stampo televisivo, cioè con un ritmo più serrato e durata intorno ai cinque minuti. Il titolo è "Satellite" (dedicato al Pioneer 10, il primo oggetto umano, a quanto è dato sapere, ad uscire dal sistema solare). A differenza di "Viaggio", il cui supporto è una videocassetta, "Satellite" utilizza direttamente i floppy disk.
Ritornando alla tua domanda, per quanto riguarda sia musica che grafica sono dell'opinione che ci siano buone possibilità sia nella pubblicità che in molte altre forme di applicazione commerciale.

G.G.

In relazione al nostro articolo "Le nuove frontiere del calcolo strutturale" pubblicato sul recente numero 38 di **Bit**, dobbiamo purtroppo informarvi della presenza di un piccolo errore in un esempio numerico proposto, errore non dovuto alla stampa, ma già contenuto nel testo da noi elaborato. Si tratta della seconda equazione del sistema a fondo della prima colonna di pagina 160: la formulazione esatta è la seguente:

$$0 + 2y - z = 1$$

anziché:

$$0 + 2y - z = 3$$

G. Forcolini

ORA C'È! ZX Spectrum

- 16 o 48 kbytes RAM.
- grafica ad alta risoluzione (256x192 punti).
- 8 colori da utilizzare con la più assoluta libertà per testo, sfondo, bordo, in campo diretto o inverso, con due gradi di luminosità, a luce fissa o lampeggiante.
- Tastiera multifunzione con maiuscole, minuscole, simboli grafici, caratteri definibili dall'utente.
- BASIC Sinclair esteso con funzioni a un tasto per programmare in fretta e senza errori.
- Funzioni specifiche per la grafica e per la gestione di dati d'archivio.
- Ampia disponibilità di programmi preregistrati su compact-cassette: giochi, passatempi, educazionali, matematici, gestionali.
- Totale compatibilità con la stampante ZX.
- Disponibilità immediata del volume ALLA SCOPERTA DELLO ZX SPECTRUM in italiano.
- Prezzo eccezionale: 360.000 lire nella versione a 16 kbytes.

sinclair ZX Spectrum FLIGHT SIMULATION — SOFTWARE BY PSION — 48K RAM

sinclair ZX Spectrum CHESS — SOFTWARE BY PSION — MICRO GEN — 48K RAM

sinclair ZX Spectrum VU-FILE also GAZETTEER — SOFTWARE BY PSION — 16K or 48K RAM

sinclair ZX Spectrum VU-CALC — SOFTWARE BY PSION — 16K or 48K RAM

sinclair ZX Spectrum VU-3D — SOFTWARE BY PSION — 48K RAM

sinclair ZX Spectrum SPACE RAIDERS — SOFTWARE BY PSION — 16K or 48K RAM

sinclair ZX Spectrum HUNGRY HORACE — PSION — MELBOURNE HOUSE — 16K or 48K RAM

sinclair ZX Spectrum PLANETOIDS also MISSILE — SOFTWARE BY PSION — 16K or 48K RAM

sinclair ZX Spectrum FUN TO LEARN SERIES MUSIC 1 — ICL — 16K RAM

sinclair ZX Spectrum FUN TO LEARN SERIES INVENTIONS 1 — ICL — 16K RAM

sinclair ZX Spectrum FUN TO LEARN SERIES HISTORY 1 — ICL — 16K RAM

sinclair è distribuito da

REBIT COMPUTER
A DIVISION OF G.B.C.

REBIT COMPUTER
Via Induno, 18
20092 CINISELLO BALSAMO
Casella Postale 10488 MI

Bitest: ICL Personal computer

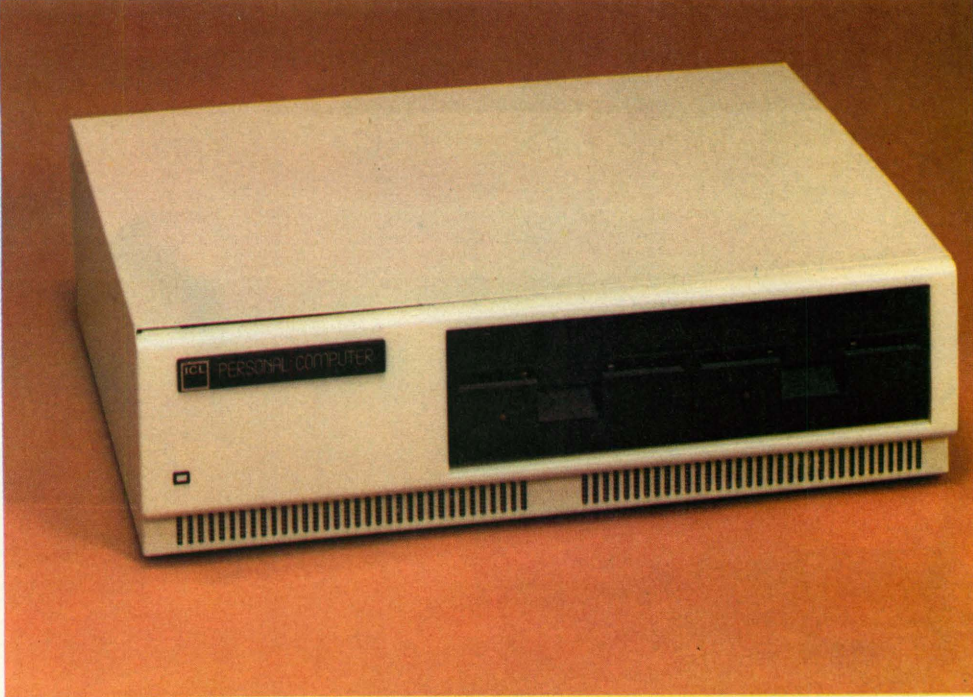
a cura della **Redazione**

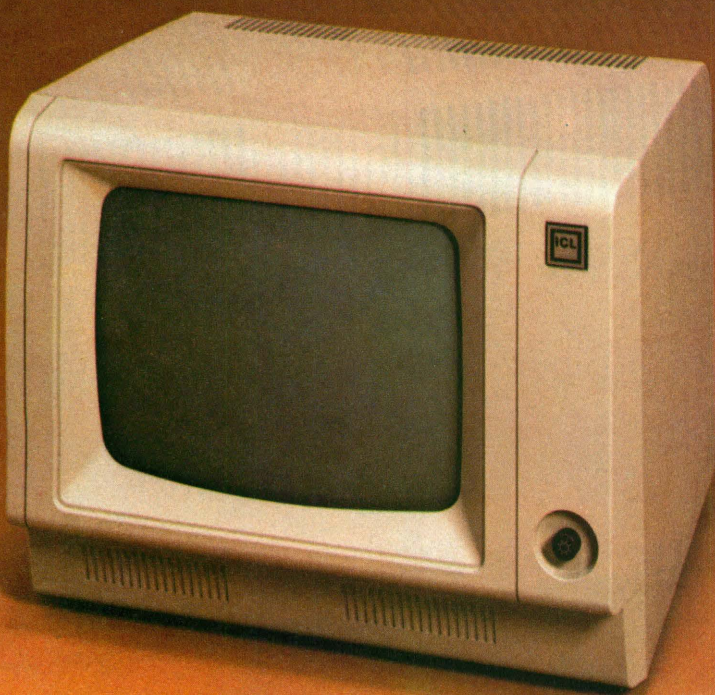
La compostezza e l'atteggiamento che sembra esprimere questa macchina. Le usuali tre unità rendono il personal computer ICL un fiore all'occhiello della progettazione inglese. Di colori sobri, il contenuto hardware di tali contenitori pare proprio ottimamente realizzato, con due processori 8085 che, l'uno nel terminale e l'altro nell'unità centrale, colloquiano in modo egregio. La multiutenza richiede buona quantità di memoria e gestione sofisticata, nonchè l'assenza di conflitti interni: un vero hardware "gentleman" rispetta le tradizioni tutte inglesi di elevata qualità tecnologica.

International Computer Limited: un altro colosso di provenienza britannica ha sfondato, già da qualche tempo, l'ormai fatidica barriera dei personal.
Sembra un punto comune di tutti i grandi complessi:creare l'utilitaria multi-funzione. Sì personal, sì intelligente, ma con una caratteristica di fondamentale utilità qual'è l'uso come terminale. Forse per l'esperienza precedente di attivo lavoro nel campo dei sistemi medio grandi, è impensabile per queste aziende non calcare le caratteristiche del proprio prodotto dal punto di vista del colloquio con host di diverse dimensioni.
È, in generale, naturalmente un ottimo punto di vista.
Utilizzare una struttura hardware già predisposta, o meglio realizzata con una filosofia che garantisca compatibilità e facilità di comunicazione, è di sicuro un buon investimento in campo internazionale. Probabilmente per il caso italiano, paese dalle mille sfaccettature problematiche a tutti ben note, l'uso terminale rimane solo un allettante beneficio per chi sogna realizzabili collegamenti a banche dati o reti informative più o meno vaste.
Come faceva notare anche la nostra redazione americana, tramite un editoriale a tutti noto apparso di recente su Informatica Oggi, il nostro paese non facilita un economico scambio di informazioni al di fuori di un rigido controllo monopolistico, altrettanto irraggiungibile dai privati, professionisti e non che siano. La comunicazione di dati è ancora ben lontana, da noi, dall'essere strumento a completa disposizione dei numerosissimi utilizzatori di personal computer.
Per questo le caratteristiche di un microcomputer come quello prodotto e commercializzato dalla ICL danno il senso della

sproporzione tra quanto è in grado di offrire un hardware neppure eccessivamente complesso, nei confronti di una organizzazione sociale totalmente incapace di dare il vero peso al fenomeno di massiccia informatizzazione che dilaga a ritmo, ormai, frenetico.
Rimane comunque la possibilità di utilizzare tali sistemi, resi così plastici sotto le mani di un attento utilizzatore da adattarsi a differenti protocolli di comunicazione, da parte di organizzazioni che vogliano decentrare l'elaborazione o la raccolta dei dati. Un uso del personal computer che andrà sempre più diffondendosi presso grosse organizzazioni di servizi che desiderano affidare ai clienti stessi parte del lavoro di digitalizzazione dei dati o della stessa elaborazione, con conseguenti riduzioni delle parcelle e contenimento delle spese.

Il personal computer ICL

Di fronte a questa macchina non si rimane, a prima vista, gran che stupiti.
L'ovvio design della classica tripletta video-tastiera-unità a dischi, non porta a immaginare nulla di particolarmente avveniristico.
L'adozione di un sistema operativo che ha, ormai, qualcosa di troppo classico, a volte un pò barocco, come il solito CP/M, non permette commenti traboccanti un nauseante entusiasmo.
Più allettante e inusuale è sentire parlare di MP/M, perfettamente operativo a tutti i livelli e ben documentato. Viene dato come sistema operativo per multiutenza, veramente realizzabile e accessibile tramite questo microcomputer, e non solo specchietto per allodole di un mercato che sempre più richiede di decentarre le unità elaborative.
Altrettanto inusuale la scelta del microprocessore: un Intel 8085 a 3 o 5 MHertz. Così viene rinfrescata la memoria di un chip non troppo diffuso, soprattutto in questo campo, ma le cui caratteristiche non hanno certo l'aspetto del banale.
La presentazione della ICL utilizza nomi diversi per ognuna delle tre unità, e ne riportiamo i termini: Personal Computer , intendendosi con ciò l'unità a dischi presentata in diferenti configurazioni e, comunque, contenente l'effettiva scheda di CPU. Il display, con ovvio significato del termine e con caratteristiche hardware tali da renderlo adatto ad uso terminale remoto e la stampante, una OKI corretta con il marchio ICL.
Da sottolineare, certi di essere di fronte ad un punto di forza, la documentazione di servizio o disponibile separatamente. Appare evidente che la ICL ha sfruttato tutta la capacità dei suoi uffici di documentazione per offrire all'utilizzatore forme chiare di

L'unità centrale comprensiva dei due dischi 5 pollici da 780 Kbyte l'uno. La semplicità rispecchia un ordine interno che a sua volta ricorda la realizzazione di hardware per sistemi ben più grandi.
La mano felice degli ingegneri della ICL ha potuto realizzare un insieme compatto e modulare, che può supportare anche dei drive per Winchester da 5 pollici, fino a capacità di 10 Mbyte.
Sembra un giocattolino, ma è invece il risultato di una felice ingegnerizzazione che, non a caso, ha scelto un microprocessore 8085 a 5 MHertz. Inusuale, ma molto intelligente per rapporto semplicità/prestazioni, specie se per risolvere la poca potenza di trasferimento dati del microprocessore in questione si è affiancato un DMA dalle caratteristiche più che notevoli: una accoppiata vincente.

Il retro dell'unità centrale con pochi elementi: l'interruttore di accensione, la presa, il fusibile, il piccolo interruttore a bilancere per il bootstrap e i quattro connettori di uscita seriale per altrettante periferiche, più o meno intelligenti.
La sua potenza sta tutta lì: capacità di dialogo contemporaneo (all'apparenza) con più utenti.
Il sistema operativo MP/M, ormai affidabile e collaudato, trova degno alloggio in questa unità centrale così semplice, ma così versatile.
Esistono altre versioni che permettono di collegare fino a 8 terminali, ed esiste anche la versione che, al posto dello sportellino centrale visibile in foto sotto l'etichetta metallica, dispone di un collegamento ai bus di sistema per ulteriori espansioni.

A vederlo così non gli si dà più di tanta importanza. L'usuale monitor a 12 pollici, fosfori verdi antiriflesso. La sua carta vincente, comunque, rimane la disponibilità piena di un 8085 a lui, e solo a lui, dedicato.
In realtà la potenza di questo terminale intelligente comincia già a esprimersi nell'alimentazione. Ovunque lo attacchiate, senza regolazioni di sorta, si adatta. Sia che siate in America, sia che siate in Europa. Così non teme neppure sbalzi di tensione anche superiori al 50%. Cosa sperare di meglio da un terminale che all'intelligenza unisce la furbizia?

manuali e di quanto altro si possa pensare utile. Si va da valigette contenenti tutta la documentazione per l'assistenza, agli audiovisivi per l'istruzione all'uso della macchina e del linguaggio, generalmente il Basic.
Viene proposto in quattro configurazioni essenziali: i modelli 15 e 25 per l'uso singolo e i modelli 26 e 35 per la multiutenza.

L'unità principale

Quella che identifica la capacità, e quindi il modello, del personal computer ICL è la sezione che contiene CPU e drive per dischi. In tutti e quattro i modelli la CPU è, come abbiamo accennato, un microprocessore 8085AH2, funzionante con un clock di 5 MHz tranne che nel mod.13 in cui viene utilizzato a 3MHz.
La scelta non è usuale nel campo dei personal. Siamo abituati a considerare il potente Z80 o il 6502, ma mai questa versione più completa dell'intramontabile 8080.
L'8085 presenta le stesse caratteristiche di microprocessore a parllelismo 8 cui siamo abitualmente portati a pensare, con un set di istruzioni identico a quello dell' 8080, fatte salve due istruzioni aggiuntive atte a gestire un ingresso e una uscita seriale pilotati da un semplice meccanismo di flag.
Con la gestione del clock entrocontenuta e i bus multiplexati è un microprocessore con delle caratteristiche simpatiche, e forse meritava una fortuna più sensibile.
L'essere scelto per gestire il personal ICL gli rende, in effetti, il giusto onore e lo eleva nella classe dei prescelti per il mercato dell'intelligenza professionale e casalinga.
La RAM centrale ha una capacità differente in base al modello.
Per l'utenza singola nei modelli 15 e 25, si limita ad un banco di 64 Kbyte, mentre arriva a 256 Kbyte nei modelli disposti alla multiutenza.
Le unità a dischi, di pari passo, crescono in capacità per gestire l'elevato volume di dati richiesti da una multiutenza crescente.
Comune a tutte e quattro le configurazioni almeno un drive per disco a 5,25 pollici, dalla capacità di '80 Kbyte data dalla gestione della doppia densità su entrambe le facce.
Poi, in ordine crescente coi modelli 15, 25, 26 e 35, sono disponibili un altro drive per dischi da 5 pollici, un disco fisso da 5 pollici 5 Mbyte, o un disco fisso da 5 pollici, ma da 10 Mbyte.
Mantenendo le dimensioni di 5 pollici è garantita la compattezza dell'insieme e l'aspetto uniforme di tutti e quattro i modelli accennati.
Per tutti, quattro porte RS 232, espandibili a 8, accettano i dati da uno o più terminali video disposti allo scopo o si collegano ad altre periferiche, quali le stampanti.
All'interno la disposizione è progettualmente più simile ad un minicomputer rilevando, a nostro giudizio, una caratteristica mentale tipica dei progettisti di sistemi di medio-grandi dimensioni.
È presente, infatti, una piastra madre che relaizza il bus di sistema e una serie di connettori che accettano fino a otto piastre verticali realizzanti le differenti funzioni hardware.
A fianco due drive, di cui uno sempre per dischi flessibili a 5 pollici, mentre l'altro per hard disk o ancora per flexidisk, variante secondo la configurazione scelta.
Sul retro, oltre le diverse prese e gli interruttori di accensione e di bootstrap, l'unità di alimentazione.
Questo è un vero "processore" di alimentazione essendo dotato di un hardware completo ed efficace che stabilizza e isola l'alimentazione dal resto dei circuiti.
Parecchi optoisolatori sono in continuo funzionamento sulle linee di feedback che tengono sotto costante controllo le uscite a bassa tensione, affinchè ogni minima variazione si ripercuota direttamente sull'alimentatore switching, che attua invece la sua regolazione direttamente sull'alta tensione.
Nel sistema è prevista anche la scelta tra frequenza americana di 60 Hz e quella europea di 50 Hz, ed in effetti la machina si adatta a qualunque situazione anche perchè, come vedremo, è disposta ad accettare caratteri alfabetici di tutto il mondo.
Un trasformatore è opportunamente schermato per evitare radiodisturbi, caratteristica di sicurezza sempre più accentuata, soprattutto nei paesi anglosassoni, dove le restrizioni in tal senso hanno limiti ben precistai e notevolmente stretti.
Le schede sono suddivise per funzione.
Ad una scheda di CPU, comprendente un DMAC, fanno seguito delle schede di RAM dinamica, una scheda per I/O seriali, un FDC e un controllore per hard disk.
La scheda dell'unità centrale di processo provvede al funzionamento del microporcessore 8085 e alla generazione del clock a 20 MHertz, suddiviso poi a dare 5 MHz o 3 MHz per il modello 15.
In questultimo caso un semplice ponticello opportunamente tagliato provvede al cambio di frequenza, e la velocità risulta così più bassa per adattarsi ai chip di minori

dimensioni da 16 Kbit, previsti nel modellino minore.
Su questa scheda è presente una PROM contenente il programma di bootstrap. Tale PROM viene abilitata ad essere letta quando l'utilizzatore preme il tasto a bilancere posto sul retro dell'unità centrale.
L'operazione attiva la lettura del sistema operativo da disco, flessibile o rigido. È previsto un tempo di "riscaldamento" di 30 secondi, dopo il quale, acceso il sistema, è possibile attivare il programma loader contenuto in siffatta PROM.
Una serie di latch permettono il demultiplexaggio dei bus, che sul micro 8085 sono uniti sulle stesse linee ed è lo stesso processore, tramite il segnale ALE (Address Latch Enable) ad avvertire della presenza dei dati o degli indirizzi sul bus stesso.
Due buffer vengono caricati e pilotati dal bus dati del microprocessore quando si usi MP/M che richiede 256 Kbytre di memoria. Con questo metodo vengono generati gli indirizzi necessari per tutti i banchi da 256 Kbyte, rendendo il bus indirizzi del sistema ampio 20 linee (0-19).
La generazione dei segnali di controllo è resa più complessa dalla presenza del chip DMA (Direct Memory Address), un 8257, che prende il posto del microprocessore stesso non appena si verifichi la necessità di uno scambio di dati tra periferiche.
È realizzato uno schema fly-by, ovvero con una effettiva rapidità di trasferimento una volta che ha indirizzato opportunamente i dispositivi, comunicanti direttamente tra loro.
Il tutto viene gestito sotto richieste di interrupt, lanciati dalle schede di controllo delle periferiche che realizzano il rimanente sistema.
È sempre questa piastra che permette l'accensione del LED rosso di bootstrap e di quello verde di sistema pronto, presenti sul pannellino frontale e ben visibili dall'operatore.
Le schede di memoria sono organizzate con RAM dinamiche da 16 Kbyte di memoria "lenta". Con quattro schede il sistema si espande a 256 Kbyte e l'esatto indirizzo del banco viene preselezionato tramite dei dip-switch presenti su ciascuna scheda, i quali agiscono da decodificatori delle linee A16-A18 del bus indirizzi, con l'aiuto di un demultiplexer da tre a otto linee.
Se lo switch sulla linea è chiuso la CPU può abilitare i buffer di quella scheda.
Gli indirizzi da A0 a A17 vengono decodificati tramite un controllore per RAM dinamiche 8828 che provvede alla generazione del segnale RAS (Row Address Signal) che permette anche il rinfresco delle celle a condensatore della memoria, e a quello CAS (Column Address Signal) che seleziona la colonna ove si trova la cella di memoria.
Per banchi da 256 Kbyte, che permettono velocità di accesso più alte e sono quindi adottati con il clock a 5MHz, sono utilizzati

Il lato posteriore è sempre una curiosità legittima dei nostri lettori. Anche in tal caso qualche particolare che denota un qualcosa di più all'interno lo si vede subito.
Fatta salva l'alimentazione, gli switch protetti da un coperchietto di plastica permettono la nazionalizzazione della tastiera, la scelta di protocolli di comunicazione e di baud rate, l'accensione o lo spegnimento del bip di tastiera e molte, molte altre combinazioni.
Il tutto, naturalmente vista la sofisticazione del terminale, può altrettanto bene essere eseguito via software.
Le prese sono presto descritte: tastiera, stampante, host computer. Quale host? Naturalmente qualsiasi, ma è certo che se per combinazione è proprio una unità centrale ICL a 5 MHertz...

L'interno del persona computer ICL dimostra con quanta cura lavorano i progettisti della International Computer Limited. Sul fondo l'alimentatore superprotetto da ogni possibile disturbo e sovratensione di rete. Si notino inoltre i due drive per dischi, coperti dalle relative logiche di controllo dei movimenti die motori e della testina.
Ad una scheda di bus sono, in questo caso, collegate altre tre schede, di cui vediamo in primo piano quella che supporta i chip di comunicazione seriale. Tale scheda è chiaramente collegata alle porte RS-232 poste sul pannellino posteriore.
In secondo piano, nell'ordine, la scheda di memoria da 256 Kbyte di cui si notano i chip con i condensatorini gialli sull'alimentazione, seguita dalla scheda della CPU che supporta il microprocessore 8085 e il chip DMA. Ultima, quella da cui parte il cavo piatto che si collega ai drive dei dischi, la scheda di controllo per i dischi stessi.
L'affidabilità dell'insieme, che comunque, come si può anche arguire dagli slot rimasti liberi, è ancora espandibile in molti differenti modi, è assicurata anche da un potente e corretto raffreddamento dovuto alla visibilissima ventola posta sul fondo.
Con tale modularità è pure assicurata una facilità estrema di intervento tecnico manutentivo, peraltro molto ben documentato su appositi manuali.

32 chip da 64 K x 1 e un controllore differente, il 2964. La selezione e le operazioni di lettura/scrittura avvengono in modo sostanzialmente identico a quanto descritto per i banchi meno capaci.
Naturalmente quattro schede di questo tipo possono espandere la RAM centrale fino a 1024 Kbyte.
In tal senso è interessante notare che tali espansioni possono essere gestite dal MP/M come un disco virtuale, ovvero trattando file direttamente in memoria e alla velocità di trasferimento elettronico, senza dover accedere alla meccanica di un drive.
In tutto e per tutto tale unità a disco virtuale viene visto come supporto magnetico, con l'unica differenza che all'atto dello spegnimento della macchina il contenuto viene perso.
Alla volatilità si ovvia, naturalmente, salvando il contenuto su un effettivo supporto magnetico (hard o flexi) con un semplice PIP.
Ciò rende estremamente comodo trattare lunghi file di dati, come nel classico esempio del Visicalc o laddove necessitino ordinamenti veloci.
Il controllore di I/O seriale è un'altra delle schede inserite nel sistema.
Ogni porta seriale, i relativi segnali di handshake e l'interrupt generato per la CPU vengono gestiti da delle USART (Universal Sinchronous Asinchronous Receiver Transmitter) presenti in numero di quattro o otto, come dicevamo in occasione delle possibilità di collegamento a 4 o 8 periferiche, tra terminali e stampanti.
La decodifica degli indirizzi è qui effettuata tramite delle PROM, opportunamente programmata in base al numero di USART presenti.
È una scheda relativamente semplice, che dà sicurezza alle comunicazioni con l'esterno.
Più impegnativa, dal punto di vista hardware, la scheda di controllo per i floppy disk.
Come molto spesso accade viene utilizzato un chip intelligente della Western Digital, il floppy disk controller FD 1793, che viene utilizzato per la gestione di due dischi doppia faccia - doppia densità.
Oltre questo chip, che sostanzialmente è in grado di assumere in toto il controllo dei dischi, sono solamente presenti delle linee di ritardo per generare i segnali con le giuste temporizzazioni e un circuito PLL che separa il clock di registrazione dai dati.
Ultima e più semplice scheda è quella che contiene la circuiteria necessaria ad interfacciare il bus di sistema con il bus richiesto dai controllori Shugart.
Si chiama SASI, da Shugart Associates Standard Interface.
È la scheda che permette l'utilizzo dei dischi rigidi. Con pochi chip contenenti semplici porte logiche, latch e multiplexer, il bus dati, quello degli indirizzi e quello di controllo della CPU viene adattato ad un tipico bus di sistema Shugart, ormai un altro degli

Dal tocco sensibile dà una gradevolezza al tatto che fa arrabbiare ogni volta che si tenta di premere i due tasti senza scritte agli spigoli a sinistra. Ci si aspetta un tocco come tutti gli altri e invece... sono solo lì per bellezza, fissi e duri come la plastica del contenitore. Interessante la possibilità di togliere dei cappucci per trasformare la tastiera stessa in un prodotto assolutamente conforme allo standard della propria nazione. La ICL ha risparmiato uniformando la tastiera e producendo diversi cappucci sostitutivi. L'utilizzatore è più contento di avere un prodotto sempre "rivendibile" a chiunque.

standard non ufficializzati che il mondo dei micro vede spesso apparire.
All'interno dell'unità centrale, facilmente smontabile tramite due semplici viti con testa a croce, è presente una robusta ventola di raffreddamento, silenziosa ma efficace.

Il terminale video

Si parla di terminale intelligente. In effetti è basato anch'esso su un microprocessore 8085 che permette di renderlo adatto a molti usi, oltre ad avere funzionalità veramente notevoli.
Il CRT è un 12 pollici a fosfori verdi, con un formato di presentazione dei dati pari a 24 righe per 80 colonne. Una 25esima riga viene dedicata a indicazioni di sistema molto interessanti. Tra queste la posizione del cursore segnalata in righe, colonne e pagine, il formato di stampa, il modo di edit linea per linea o a pagine, il modo grafico o l'intensità dei caratteri, il baud rate selezionabile da 75 a 9600 baud, i messaggi d'errore e altre segnalazioni particolarmente utili.
Il cambiamento dei parametri che determinano i protocolli di comunicazione con l'host computer e la stampante è effettuabile tramite degli switch posti sul retro del sistema.
Tramite questi ultimi è possibile scegliere gli 8 differenti baud rate, la parità da assegnare, la frequenza di rete a 50 o 60 Hz, il bip associato alla tastiera, la porta RS232 o current loop, il linguaggio in cui si desidera scrivere (quindi la configurazione della tastiera stessa), il video totalmente in campo inverso.
Ancora, è selezionabile la trasmissione dati in half o full duplex, in modo a blocchi o l'uso locale.
Il retro del video, oltre gli interruttorini detti, monta il fusibile, l'interruttore e la presa di alimentazione.
A tale proposito è interessante notare che l'alimentatore, scelta la sola frequenza di rete, è in grado di operare automaticamente e senza nessuna preliminare selezione, tra 96 e 264 V C.A.
Quindi potete girare il mondo o avere i più pesanti sbalzi di rete, ma sarete sempre certi di un perfetto e continuo funzionamento.
Ancora, un forellino mette in grado l'utilizzatore di raggiungere il trimmer di regolazione del volume del bip asservito alla pressione dei tasti.
Per il collegamento con la tastiera esiste apposita presa, e così anche per la stampante e l'host-computer (tipicamente l'unità centrale ICL appena descritta).
Di fronte una manopolina regola il contrasto.
L'aspetto complessivo poteva essere forse più adatto ad un design moderno, meno rispecchiante l'austera Inghilterra...
La descrizione completa di questo sofisticato terminale necessiterebbe di molte pagine, almeno pari al voluminoso manuale, non unico, che ne descrive ogni recondita possibilità.
Sono disponibili una notevole quantità di caratteri, compresi gli inversi e i sottolineati.
È anche possibile selezionare il set grafico che, seppure non a punti, ma evidentemente non ne è previsto un uso così particolare, contiene una serie completa di simboli.

La tastiera

È un elemento indiscutibilmente unito al terminale video. Non solo fisicamente, tramite un cavetto di comando lunghezza, ma anche e soprattutto logicamente.
È la tastiera, evidentemente, che permette di lanciare al terminale quelle sequenze di caratteri, usualmente preceduti da ESC, che permettono di selezionare i differenti modi di funzionamento.
Così, ad esempio, premendo ESC e Shift ogni comando verrà interpretato in modo locale, senza essere inviato all'host computer. Tutto, dalla seslezione del baud rate a quella dei protocolli di trasmissione, può essere selezionato via software, inviando l'opportuna sequenza di caratteri.
Esteticamente appare tipicamente a basso profilo, con tastierino numerico separato (e duplicazione dell'ENTER), e una serie di tasti funzione riservati all'utente e al sistema, per un veloce data entry o selezione di modi di funzionamento.
Presenta dei tasti molto morbidi, che rendono il tutto molto sensibile, quindi di uso veloce al professionista della battitura. Per l'utente usuale, colui il quale non sa digitare se non con due dita abilmente spostate in lungo e in largo, la maggiore sensibilità al tocco si traduce regolarmente in un eccesso di errori di doppia battuta, ma è solo questione di abitudine.
I progettisti ICL dimostrano l'intelligenza della realizzazione anche tramite un particolare del tutto originale. Sono infatti forniti con il sistema tutti i tasti diversi dallo standard britannico, così è possibile, che si scriva in italiano o in svedese, personalizzare la propria tastiera secondo la nazionalità.
Un utile attrezzo rende banale tale sostituzione, che chiamare hardware parrebbe eccessivo, ma che in effetti è un ottimo trucco nato da persone furbe.
Tutte le configurazioni possibili della tastiera sono chiaramente descritte e disegnate in un apposita sezione del manuale d'uso, rispettando ancora una volta l'impressione di completezza che la ICL ha voluto dare all'utente.

La stampante

A tutti ormai ben nota, una OKI 83A è stata siglata ICL e viene venduta come stampante di sistema.
Con 120 cps, scelta del set di caratteri, semigrafico compreso, diverse spaziature orizzontali e verticali, si mantiene nello standard medio-alto delle stampanti a 132 colonne.
Un perfezionamento che, speriamo, la OKI dovrebbe prendere in considerazione è la forma e la disposizione del microswitch di segnalazione fine-foglio, quando si usi il caricamento dal retro del rullo.
È una linguetta metallica con la pessima caratteristica di strappare la carta se solo si voglia arretrare il rullo stesso. L'unico rimedio è utilizzare il caricamento tramite le fessure sul fondo, cosa impossibile se si sta utilizzando la stampa a foglio singolo.
Non potrebbero intervenire gli ingegnosi progettisti ICL?...

Il software e la documentazione

Come abbiamo necessariamente detto parlando dell'hardware, l'elaboratore supporta due sistemi operativi, l'uno dedicato all'utenza singola, l'altro alla multiutenza.
Sotto l'ombrello CP/M e MP/M sono disponibili schiere di programmi applicativi, e chiunque, con un buon Basic 80, è in grado di cimentarsi in programmi ad hoc per le proprie esigenze.
Fino a che la ICL non presenterà, e pare bene intenzionata, una scheda CPU a 16 bit con 8088 , l'utilizzo dell'uno o dell'altro dipende solo dal modello di computer scelto, ovvero dall'estensione di memoria disponibile.
Dovendo gestire gli hard disk è previsto un comando di back-up che copia totalmente il Winchester sul floppy-disk, utilizzando un formato non standard che compatta i dati nel miglior modo possibile.
È inoltre capace di salvare solo i file aggiornati più di recente, lasciando intatto il resto.
Così risulta spresso una operazione veloce, cosa gradita soprattutto quando si avrebbero da copiare 10 Mbyte.
Una cache-memory presente sui sistemi dedicati alla multiutenza rende ancora più veloce un accesso ai dati del disco rigido. Viene, infatti, fatta una copia dei dati del disco più prossimi alla zona ove è avvenuto l'ultimo accesso. Così è presumibile che ogni utente trovi subito i propri dati su RAM invece di doverli andare a cercare ogni volta su un disco che un altro utente potrebbe avere impegnato proprio in quel momento. È molto difficile razionalizzare tale discorso, perchè si basa sulla probabilità che i dati necessari siano proprio quelli caricati nella cache-memory. D'altronde è

Costruttore:

ICL International Computer Limited
ICL House, Putney London SW 15 15 W

Distributore:

ICL Italia S.p.A.
Centro Direzionale Milanofiori
20094 MILANO

Listino prezzi:

Articolo	Prezzo
ICL Personal Computer - Mod. 15 64 Kbyte RAM - 2 unità per minifloppy da 800 Kbyte - CP/M - BASIC 80	*L. 5.300.000+IVA*
ICL Personal Computer - Mod. 25 64 Kbyte RAM - 1 unità per minifloppy da 800 Kbyte - 1 hard disk da 5 Mbyte - CP/M - BASIC 80	*L. 10.080.000+IVA*
ICL Personal Computer - Mod. 26 256 Kbyte RAM - 1 unità per minifloppy da 800 Kbyte - 1 hard disk da 5 Mbyte - MP/M - BASIC 80	*L. 11.040.000+IVA*
ICL Personal Computer - Mod. 35 256 Kbyte RAM - 1 uità per minifloppy da 800 Kbyte - 1 hard disk da 10 Mbyte - MP/M - BASIC 80	*L. 12.360.000+IVA*
Stampante 3181	*L. 1.212.000+IVA*
Stampante 3184	*L. 1.788.000+IVA*
Stampante 3185	*L. 2.376.000+IVA*

dimostrato che si risparmia un tempo sensibile, e che la probabilità accennata non è poi così infima.

A livello software viene direttamente gestito quello Micropro, tra non molto personalizzato anche in Italiano.

Sforzi di traduzione sono anche rivolti alla documentazione, per ora solo in inglese, ma estremamente chiara e ben fatta, oltrechèvoluminosa.

All'utente vengono forniti insieme alla macchina la guida per l'utente e il manuale del CP/M o MP/M.

Sono comunque disponibili anche molti altri volumi, tra cui spiccano i manuali d'assistenza e quelli tecnici, che guidano passo dopo passo alla soluzione dei problemi hardware e danno dettagliatissima descrizione di ogni parte, meccanica e elettronica, di tutti i modelli.

Una vera miniera di schemi di progettazione, di elenchi di parti di ricambio, di diagrammi di flusso per la ricerca guasti, di consigli di manutenzione.

È anche questa dovizia di documentazione che dà l'immediata impressione di una macchina nata non "a caso", ma pensata da un gruppo di progettisti ben guidato da una azienda che si muove uniformemente.

Interessanti i corsi di Basic su audiocassette e di conoscenza del CP/M, ancora su audiocassette.

Addirittura si può essere introdotti all'uso del sistema e alla sua manutyenzione tramite delle videocassette, sempre in inglese, ma presto tradotte.

Molti manuali sono raccolti in spessi e eleganti raccoglitori plastici, che per comodità di lettura sono apribili a leggio. I disegni abbondano e la disposizione dei divisori, degli indici, dei diagrammi di consultazione denotano uno studio di manualistica degno di sistemi di ben altre dimensioni.

Non per nulla, crediamo, la ICL ha voluto fornirci per questo Bitest un intero scatolone di sola documentazione, la qual cosa non è usuale, ma effettivamente merita di essere vista e presa ad esempio.

Conclusioni

Molto difficile porre delle considerazioni assolute: di fronte a questo sistema ci si sente pervasi esattamente dalla mentalità inglese. Non brillante in estetica (ma quale sensazione pervade chi visita l'Inghilterra se non quella d'un antico leggermente triste) è un ottimo prodotto dal punto di vista tecnico e progettuale. Sicuramente robusto, destinato a durare e comunque così ben documentato che diventa quasi facile rimediare ad eventuali guasti seppure reconditi.

L'affidabilità, pare il suo pregio.

Se a questo uniamo la multiutenza e la dovizia di documentazione di supporto non possiamo che affermarne l'uso più corretto in ambito aziendale e professionale.

Come, per fortuna, siamo propensi a dire da un pò di tempo a questa parte, anche questa è un'ottima macchina.

Buon segno. Il mercato dei personal sta dando segni di buona maturità. ■

di P. Umiliacchi

Parte seconda

Teoria delle operazioni

Proseguendo nella descrizione delle caratteristiche funzionali dello Standard IEEE-488, esaminiamo ora le diverse operazioni da esso previste. Esse sono qui sotto elencate:

- indirizzamento,
- azzeramento dei dispositivi,
- programmazione a distanza o locale,
- sincronizzazione,
- richiesta di servizio,
- interrogazione seriale,
- interrogazione parallela,
- passaggio del controllo.

Occorre tenere presente che non sempre tutte queste possibili funzioni operative sono implementate in una interfaccia e non sempre in maniera completa. Lo Standard prevede infatti opzioni che possono essere trascurate in sede realizzativa, senza pregiudicare la compatibilità tra le diverse interfacce (possono però non essere possibili alcune operazioni). Per maggior comodità di chi legge, nella Tav. 1 si riportano le linee di segnalazione ed i segnali di comando in gioco nello Standard.

Indirizzamento

All'atto dell'accensione del sistema tutte le interfacce vengono azzerate dal messaggio *IFC* inviato dal controllore di sistema, che pone a livello basso la corrispondente linea del bus e diventa controllore in carica. A questo punto tutte le interfacce sono pronte a ricevere comandi e indirizzi, mentre nessun collegamento è stabilito tra i dispositivi che tali interfacce collegano al bus. Il controllore in carica provvede a predisporre tale collegamento indirizzando uno o più ascoltatori e un solo parlatore. La regola è infatti che non più di un parlatore per volta possa essere attivato, come risulta logico da quanto si è detto la volta scorsa: lo standard mira ad evitare ogni conflitto sul bus. La cosa è garantita ulteriormente dal fatto che, quando viene inviato sul bus l'indirizzo di un parlatore, qualora sia attivo un altro parlatore precedentemente indirizzato, questoautomaticamente si disimpegna dal bus.

Ogni interfaccia cui perviene il proprio indirizzo, tiene conto di questo fatto memorizzandolo, però non stabilisce ancora la connessione tra il dispositivo ad essa relativo e il bus. Perché ciò avvenga è necessario che il controllore si disattivi, andando nello stato di *stand-by* o di attesa. L'indicazione di tale situazione è rappresentata dal fatto che la linea ATN va a livello alto (non attiva). È soltanto a questo punto che le interfacce stabiliscono il collegamento tra i relativi dispositivi i quali, per loro tramite, possono così comunicare, inviando sul bus comandi di programmazione, risultati di misure o dati di altra natura. Terminata la trasmissione, il parlatore pone *EOI* a livello basso (oppure trasmette un byte che indica la fine del messaggio) segnalando al controllore che può riprendere il controllo, riattivando *ATN*. Comunque il controllore può in qualsiasi momento riprendere il controllo, attivando *ATN*, il che provoca l'immediata cessazione di ogni attività da parte delle interfacce, che si pongono nello stato di attesa di nuovi comandi.

La configurazione dei collegamenti sul bus può così essere cambiata dal controllore, vero e proprio arbitro indiscusso delle comunicazioni. Occorre tener presente che indirizzando un nuovo ascoltatore, esso *si aggiunge* a quelli già indirizzati, per cui l'unico modo di modificare il gruppo di ascoltatori è quello di disattivarli tutti mediante il comando *UNL* e poi indirizzare solo quelli voluti. Ciò non è necessario per cambiare il parlatore, come si è detto.

Il comando *UNT* serve pertanto solo quando non si vuole avere nessun parlatore. Uno schema riassuntivo semplificato (che non riproduce questi ultimi dettagli) è riportato nel grafo di figura 1.

Una eccezione a questo modo di operare si ha nel caso di un sistema minimo che compren-

Oltre a strumenti di misura veri e propri, il vasto assortimento di apparecchiature IEEE-488 compatibili comprende anche periferiche di ogni tipo. Qui è mostrato un plotter PM 8151 della Philips.

Linea	Dispositivo	Significato
IFC REN	Controllore di sistema	Reset interfacce Programmaz. locale/remota
DAV	Parlatore	Dato valido
NRFD NDAC	Ascoltatore	Non pronto a ricevere Dato non accettato
SRQ	Parlatore Ascoltatore	Richiesta di servizio
EOI	Parlatore Controllore	Fine dati o interrogazione
ATN	Controllore	Attenzione! Controllore attivo

COMANDI			
Universali		**Indirizzati**	
UNL	Unlisten	GTL	Go to local
UNT	Untalk	GET	Group execute trigger
LLO	Local lock out	PPC	Parallel poll configure
DCL	Device Clear	PPE	Parallel poll enable
PPU	Parallel poll unconfigure	PPD	Parallel poll disable
SPE	Serial poll enable	TCT	Take control
SPD	Serial poll disable		

Tavola 1 - Riepilogo delle linee di segnalazione (sopra) e (sotto) dei vari comandi "universali" e "indirizzati" in gioco nello standard IEEE-488.

Note sullo standard IEEE-488

delle apparecchiature sul bus, mentre non ha effetto sui dispositivi collegati ad esse. Poiché può essere necessario assicurarsi che anche i dispositivi si trovino nella loro condizione iniziale, all'accensione o in seguito a manovre errate, è previsto il comando *DCL*, che azzera *tutti* i dispositivi presenti sul bus, essendo un comando universale. Se invece si vuole resettare solo uno o alcuni dei dispositivi collegati, si deve prima indirizzarli come ascoltatori, quindi inviare il comando *SDC*.

Cavo di collegamento tra apparecchiature dotate di interfacce IEEE-488. La lunghezza di tali cavi va da 1 a 4 metri.

da solo un parlatore e un ascoltatore. Lo Standard consente la comunicazione del parlatore all'ascoltatore prevedendo per il primo uno stadio di "*talk only*", in cui l'apparato è sempre indirizzato come parlatore, e per il secondo uno stato di "*listen only*", cioè di indirizzamento permanente come ascoltatore. L'apparato entra in uno di questi due stati mediante un apposito switch. Si ha, cioè, una sorta di "indirizzamento locale" che consente di fare a meno del controllore.

Reset dei dispositivi

Il messaggio *IFC*, inviato dal controllore di sistema, provoca un reset delle sole interfacce

Programmazione a distanza o locale

Quando un dispositivo deve essere programmato dal controllore tramite il bus, il controllore di sistema (e solo lui) può attivare la linea *REN*. L'effetto ovviamente si esercita su tutte le apparecchiature, ma non ha carattere immediato. Infatti il comando *REN* diventa operante su ogni apparecchio solo dopo che l'apparecchio stesso è stato indirizzato come ascoltatore. In qualche momento può però essere in seguito necessario l'intervento umano, per cui si deve riabilitare la possibilità di controllare l'apparecchio tramite il pannello frontale. Il controllore in carica può allora indirizzare tale apparecchio come ascoltatore e inviare il comando *GTL*. Oppure può essere l'operatore stesso a premere il pulsante *LOCAL* sul pannello frontale. Entrambi questi comandi non hanno però effetto se precedentemente il controllore ha inviato il comando *LLO*. In questo caso l'unico modo per far tornare gli apparecchi nella modalità locale è quello (sempre valido) di disattivare la linea *REN*.

Trigger - sincronizzazione

In taluni casi è necessario far sì che due o più dispositivi inizino ad operare contemporaneamente, oppure che un dispositivo inizi ad operare in un istante ben preciso.
Ciò può essere ottenuto indirizzando tali apparecchi come ascoltatori e inviando il comando *GET*. Alla ricezione di tale comando ogni dispositivo inizierà immediatamente a compiere l'operazione precedentemente programmata.

Richiesta di servizio

In condizioni normali una generica apparecchiatura collegata al bus può comunicare con il controllore solo quando è quest'ultimo a deciderlo. Si possono tuttavia presentare situazioni che richiedono un interessamento più sollecito da parte del controllore, come nel caso di allarmi, indicazioni di misura pronta, difetti di funzionamento e così via. Quando una apparecchiatura si trova nella necessità di informare con urgenza il controllore di una particolare situazione, può farlo attivando, a livello basso, la linea *SRQ*. Questa linea è collegata a tutti gli apparecchi, che al solito la pilotano con uscita di tipo open-collector. Pertanto essa si attiva quando almeno un apparecchio pone *SRQ* bassa.
Il controllore si accorge allora della richiesta di servizio, senza però sapere quale apparecchiatura l'ha inviata. La funzione della linea *SRQ* è simile a quello di una linea di interrupt nel caso di un microprocessore, senza però che il controllore sia tenuto a rispondere immediatamente ad una sua attivazione.

Interrogazione seriale

Per individuare l'apparecchiatura che sta richiedendo servizio, il controllore che ha riscontrato l'attivazione di *SRQ* dispone di due modi. Il primo consiste nell'interrogare serialmente tutti gli apparecchi, facendosi trasmettere il loro byte di stato. A tal fine il controllore blocca ogni attività sul bus e invia il comando *SPE*, che attiva il "polling" seriale. Quindi si pone ad ascoltare i diversi apparecchi che vengono indirizzati uno alla volta in qualità di parlatori. Essi infatti, in risposta al loro indirizzamento come parlatori, inviano il proprio byte di stato. Gli apparecchi che hanno richiesto servizio (con l'*SRQ*) pongono il bit 7 di tale byte a 1, consentendo così la loro identificazione da parte del controllore. Inoltre gli altri bit del byte di

stato possono servire a comunicare ulteriori informazioni relative allo stato del dispositivo, al tipo di servizio richiesto, ecc. Terminata l'interrogazione di tutti gli apparecchi, il controllore invia il comando *SPD*. A causa del suo procedere sequenzialmente, l'interrogazione seriale è piuttosto lenta, ma per contro ha il vantaggio di informare il controllore sullo stato di ogni dispositivo interrogato.
È importante sottolineare che l'apparecchio che ha inviato richiesta di servizio, una volta che ha risposto all'interrogazione seriale, smette di mantenere attiva la linea *SRQ*.

Interrogazione parallela

Un metodo più rapido per identificare l'apparecchio che ha richiesto servizio, consiste nell'interrogare *contemporaneamente* tutti gli apparecchi. Questa procedura in parallelo richiede che ogni apparecchio sia stato preventivamente predisposto a rispondere a questo tipo d'interrogazione. La cosa può essere attuata localmente a cura dell'operatore, mediante appositi switch oppure, a distanza, dal controllore. La predisposizione consiste nell'assegnare all'apparecchio una delle 8 linee dati, sulla quale dovrà rispondere, in caso abbia richiesto servizio, ponendola alta o bassa. In totale sono necessari 4 bit per codificare tali informazioni (3 per la scelta della linea e 1 per il tipo di risposta). Quindi si avranno 4 switch sull'apparecchio oppure i 4 bit sono inviati dal controllore in questo modo.
L'apparecchio viene prima indirizzato come ascoltatore, quindi il controllore invia il comando *PPC*. A questo segue il comando *PPE*, i cui 4 bit meno significativi sono quelli corrispondenti alle informazioni di cui sopra. Il comando *PPE* (come quello che sarà visto in seguito, *PPD*) è un comando *secondario*, ossia può essere capito solo se preceduto dal comando *PPC*.
Questa operazione deve essere eseguita su tutti gli apparecchi che debbono rispondere all'interrogazione parallela.
A questo punto l'attività sul bus può procedere normalmente. In ogni istante, però, il controllore in carica può eseguire una interrogazione parallela, operazione che compie attivando contemporaneamente le linee *ATN* e *EOI*. Ciò provoca una sospensione dell'attività delle interfacce, che rispondono sulla linea prevista, con il livello assegnato, se stanno richiedendo servizio.
Il controllore può così interrogare, individuandoli di colpo, sulla base di quel bit, unico tra 8, attivo secondo i "patti" preliminari, fino ad 8 apparecchi. Si possono anche assegnare più apparecchi ad una stessa linea. Fatto ciò il controllore disattiva *ATN* e *EOI* e l'attività sul bus riprende dal punto in cui era stata interrotta. L'interrogazione parallela non richiede indirizzamenti o trasmissione di comandi sulle linee dati. La risposta segue quasi immediatamente la segnalazione del controllore. Se a ciò si aggiunge che si interrogano contemporaneamente più apparecchi, si capisce come questa procedura sia molto più veloce della precedente. Tuttavia, a differenza di quanto capita con l'interrogazione seriale, quella parallela non fornisce al controllore alcuna informazione circa il tipo di servizio richiesto. La configurazione degli apparecchi per l'interrogazione parallela può essere annullata indirizzando ogni apparecchio e in-

Figura 1 - Il grafo, semplificato ed incompleto, illustra la sequenza principale relativa alle modalità dell'indirizzamento. La linea di transizione tratteggiata sta ad indicare la possibilità, da parte del controllore , di interrompere in qualsiasi momento il dialogo tra "parlatore" e suoi "ascoltatori": sia aggiungendo nuovi ascoltatori sia modificando il parlatore. Per maggiori dettagli si veda il testo.

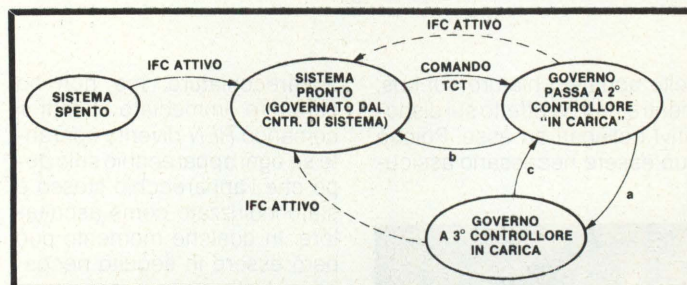



Figura 2 - Il grafo schematizza la gerarchia dei passaggi del governo dal controllore di sistema ad eventuali controllori subordinati, che assumono il governo "in carica" su delega di quelli di rango maggiore. Le linee contrassegnate con a, b, c, corrispondono a transizioni di cedimento spontaneo del controllo a controller gerarchicamente più elevati. Le linee tratteggiate indicano invece la possibilità che il controllore di sistema, in qualsiasi momento, riprenda in mano il governo "d'autorità", attivando la linea IFC.

Figura 3 - Esempio di circuito elettrico che consente il collegamento al bus (in questo caso bidirezionale). Un tale circuito dovrà essere presente per ognuna delle linee del bus che vengono utilizzate dall'interfaccia.

Figura 4 - Connettore e collegamento dei segnali secondo lo Standard IEEE-488. Si hanno in tutto 24 linee.

stampanti STAR prestazioni e prezzi eccezionali

Le nuove STAR 510 e 515 rappresentano l'ultimo in ordine di tempo e più eccezionale risultato della STAR nel settore delle stampanti a impatto.
La qualità di stampa, le capacità grafiche, la robustezza e l'affidabilità sono le caratteristiche più importanti.
I modelli 510 a 80 colonne e 515 a 132 colonne per prestazione e prezzo lanciano la sfida a quanto di meglio è disponibile sul mercato per la stampa dati e grafici in collegamento al computer.
Le DP 510 e DP 515 sono stampanti ad impatto ad aghi a matrice 9x9.
La velocità di stampa è di 100 cps. con "throughput time" di 48 Lpm, la stampa è bidirezionale a percorso ottimizzato.
Altre caratteristiche sono bit image ad alta risoluzione (120x144), un set di caratteri grafici a matrice 6x6, il basso livello di rumore

Pronta consegna.

Note sullo standard IEEE-488

viando il comando *PPC* seguito dal comando *PPD*. Se tutto ciò deve essere fatto per tutti gli apparecchi, ci si può limitare all'invio del comando universale *PPU*.

Passaggio del controllo

Consideriamo da ultimo il caso che siano collegati al bus più controllori. All'accensione del sistema, come si è visto, il governo del bus è assunto dal controllore di sistema, il quale

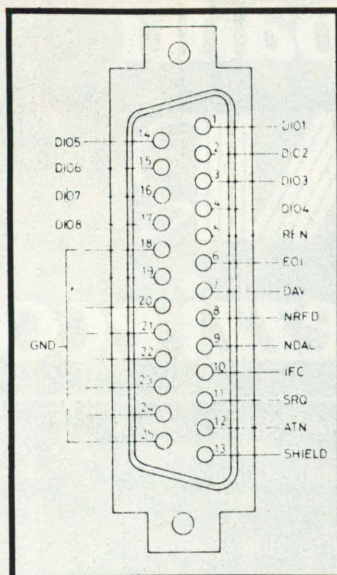



Figura 5 - Connettore e collegamento dei segnali secondo lo Standard IEC-625. In questo caso le linee sono 25 (c'è una linea di massa più).

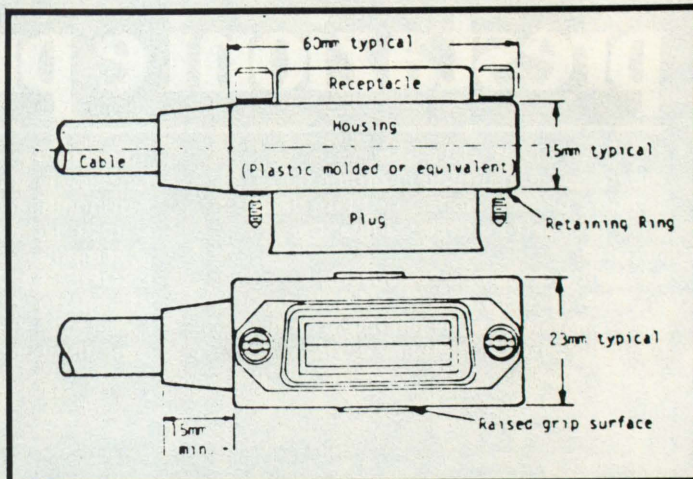



Figura 6 - Il connettore volante che termina il cavo di collegamento, secondo lo Standard IEEE-488, incorpora in realtà due connettori, un maschio (plug) e, dalla parte opposta, una femmina (receptacle).

COMANDO	ASCII	HEX	DEC
UNL	?	3F	63
UNT	—	5F	95
LLO	(DC1)	11	17
DCL	(DC4)	14	20
PPU	(NAK)	15	21
SPE	(CAN)	18	24
SPD	(EM)	19	25
SDC	(EOT)	04	4
GTL	(SOH)	01	1
GET	(BS)	08	8
PPC	(ENQ)	05	5
PPE		6X	96+Y
PPD		70	112
TCT	(HT)	09	9

NB: I 4 bit meno significativi del codice relativo al comando PPE sono variabili in quanto servono a programmare la risposta dell'interfaccia alla parallel-poll.

Tavola 2 - Tabella dei comandi.

segnala ciò attivando la linea *IFC* per un periodo non inferiore a 100 μsec.
In seguito esso può passare il controllo ad un altro dei controllori presenti. Per farlo deve prima indirizzare tale controllore mediante l'indirizzo del parlatore ad esso associato, quindi inviare il comando *TCT* (Take Control). Il ruolo di controllore attivo passa così al nuovo controllore, che assume il pilotaggio della linea *ATN* ed è abilitata all'invio di comandi e indirizzi.
Con analogo procedimento il controllore in carica può passare a sua volta il governo ad un terzo controllore come pure tornare a cederlo al controllore di sistema.
Quest'ultimo però rimane sempre in posizione privilegiata gerarchicamente, dato che in qualsiasi momento ha la facoltà di riprendere in mano le fila del "potere" inviando nuovamente il comando *IFC* e bloccando così qualsiasi attività sul bus.
Il tutto è schematicamente riassunto nel grafo di figura 2.

Caratteristiche elettriche

Il collegamento elettrico dell'interfaccia al bus deve avvenire mediante appositi circuiti, pilota e ricevitori, basati sulla tecnologia TTL. Poiché il bus è in logica negativa, i due livelli logici sono:
livello alto valore $0 \geqslant 2V$
livello basso valore $1 \leqslant 0,8V$
Quindi i ricevitori debbono essere in grado di riconoscere questi livelli assicurando un certo margine contro il rumore.
I circuiti pilota debbono essere di tipo open-collector, almeno per quanto riguarda le linee *SRQ, NRFD, NDAC* e le 8 linee dati se si usa la parallel-poll. Le altre linee possono essere anche three-state, soprattutto se sono richieste velocità di trasmissione elevate. Il pilota deve essere in grado di assorbire 48 mA mantenendo un livello basso inferiore a 0,4V e di fornire 5,2 mA ad un livello alto superiore a 2,4V. Viene così assicurato un margine di immunità al rumore, nelle condizioni peggiori, di 400 mV, come nella TTL standard. Ogni linea del bus, all'interno di ogni dispositivo, deve terminare con un partitore resistivo, che ha lo scopo di fissare il livello della linea quando tutti i pilota sono inattivi e di aumentare l'immunità al rumore. Per i ricevitori deve poi anche essere presente un diodo, con lo scopo di evitare la possibilità di livelli di tensione molto negativi, come possono aversi a causa delle brusche escursioni dei segnali e del carico capacitivo della linea. Quest'ultimo non deve superare 100 pF per ogni linea.
Queste considerazioni sono oggi spesso superflue, esistendo in commercio circuiti integrati che comprendono non solo i circuiti pilota e ricevitori (driver bidirezionali) ma anche le resistenze di terminazione e i diodi, il tutto per 4 o anche 8 linee. Per maggior completezza si riporta comunque nella figura 3 il circuito relativo ad una configurazione tipica.

Caratteristiche meccaniche

Come è noto, uno dei problemi che più di frequente si presentano nell'interconnessione di apparati diversi è quello dei connettori, la cui standardizzazione riveste importanza notevole dal punto di vista pratico. Purtroppo è proprio nel connettore che la normativa internazionale (IEC-625) si è scostata da quella originaria scelta dell'IEEE-488, per cui si possono avere problemi collegando apparecchiature americane con altre europee.
Per lo standard IEEE-488 il connettore deve essere a 24 poli, con polarizzazione trapezoidale e l'assegnazione dei contatti è fatta secondo lo schema di figura 4.
Per lo standard IEC-625 invece il connettore deve *essere a 25* poli, tipo Canon DB-25 (lo stesso previsto dallo standard RS 232-C). Esso è visibile nella figura 5.
Si avranno poi connettori femmina montati sul pannello posteriore di ogni apparecchiatura, mentre i cavi di collegamento (di lunghezza massima pari a 4 m.) termineranno con connet-

Harden Italia. Il salto di qualità.

Dal personal computer al professional computer.

Nel quadro di una filosofia aziendale in evoluzione, Harden Italia riconferma la validità della proposta del Sirius 1. Il Sirius 1, con tutta la potenza del suo microprocessore a 16 bit, con 5 MHz, e una memoria centrale che può arrivare 896 KBytes, è uno dei più avanzati della nuova generazione dei Personal.

Oltre ad una enorme capacità di archiviazione dei dati (dai 1240 KBytes del Sirius 1 agli 11.840 KBytes del Sirius 1b) il Sirius può contare su alcune caratteristiche che un tecnico e un professionista non possono non apprezzare: dall'interfacciamento con due porte seriali e una parallela programmabile da software, ai sistemi operativi (MS-DOS della Microsoft e CP/M86 della Digital Research), fino ai linguaggi di alto livello come il BASIC-86 (interprete e compilatore), l'Assembler, il COBOL, il Fortran, il Pascal.

Oltre che sul software vero e proprio (programmi come il Dbase II, il SuperCalc, il Multiplan o l'Harden-text e l'Harden-data) il Sirius 1 si avvale dei così detti "Tool Kits", una serie cioè di utilities compatibili con qualsiasi linguaggio che permettono una stesura dei programmi più facile e più completa come ad esempio l'AutoSort, il FABS, una gestione sofisticata IS, ecc.
In più, il Sirius 1 è distribuito e assistito dalla Harden Italia su tutto il territorio nazionale.

Per saperne di più sul Sirius 1, sui suoi programmi o su dove sono i punti di vendita Harden più vicini, chiamare (0372)-63136 oppure (02)-651645: risponde la Harden Italia.

sirius

Harden Italia S.p.A. Direzione generale e uffici commerciali
20121 Milano - via dei Giardini, 4 - tel. (02) 651645
Sede operativa e uffici commerciali
26048 Sospiro (CR) - tel. (0372) 63136 - telex: 3205881

SIRIUS 1 CONFIGURAZIONE BASE
(128 KBYTES RAM, 1240 KBYTES FLOPPY DISC)
DA OGGI L.6.500.000

Job Line

Note sullo standard IEEE-488

tori maschi.
La norma IEEE-488, inoltre, prevede che i connettori terminali di cavi siano costituiti da un maschio unito ad una femmina in un unico elemento (figura 6). Questo sistema consente di collegare più apparecchi in configurazioni a stella o di prolungare il collegamento verso nuovi apparecchi. Qualunque tipo di collegamento, anche misto, è consentito. La lunghezza massima complessiva deve però essere limitata da un numero di metri pari al doppio del numero di strumenti collegati, con un massimo assoluto di 20 metri. Quindi, ad esempio, due apparecchi possono essere collegati con un cavo di lunghezza massima pari a 4 metri.

Bibliografia

La bibliografia relativa allo Standard IEEE-488 è andata sempre più ampliandosi, col passare degli anni. Pertanto elenco solo qualche titolo di riferimento che ho consultato per questo articolo.

1) IEEE 488-1978 "Standard Digital Interface for Programmable Instrumentation" reperibile presso Edelektron.
2) IEC n. 625-1980 "Interface System for Programmable Measuring Apparatus, Byte-Serial Bit parallel".
3) Donald C. Loughry, Mark S. Allen "IEEE Standard 488 and Microprocessor Synergism" - Proceedings of the IEEE, no. 2 Febbraio 1978.
4) Paolo Schiaffino "Sistema di interfaccia per apparecchi di misura programmabili" - L'Elettrotecnica - Febbraio 1980.
5) Motorola "Getting Aboard The 488 - 1975 BUS".
6) Hewlett-Packard "HPIB - Versatile interconnect system for instruments and controllers".
7) Andy Santoni "IEEE-488 Compatible instruments" EDN - Novembre 1979.
8) FLUKE "IEEE-Standard 488-1978 Digital Interface for Programmable Instrumentation" - Application Bulletin AB-36.
9) Eugene Fisher - C.W. Jensen "PET and the IEEE-488 bus (GPIB)" Osborne/Mc Graw-Hill.
10) "Metodi di interfacciamento ed Interfacce Standard nei Sistemi Elettronici" - Edelektron.

■

Ma ora si ha pure il governo a distanza

La principale limitazione del Bus IEEE 488 è che con la sua versione tradizionale non consente l'accesso ed il governo a distanza degli strumenti, il che oggi può considerarsi un fatto serio, col crescente affermarsi dei sistemi distribuiti in genere e di quelli remoti in particolare.
Rivisitando superficialmente il meccanismo di handshake del Bus, trattandosi di trasferimento di tipo asincrono può a prima vista apparire che la lunghezza dei cavi possa aumentare a piacere, pur con la conseguenza di una sempre più ridotta della velocità di trasferimento. In realtà operazioni essenziali del controller prevedono una risposta degli apparecchi allacciati entro una finestra temporale ben definita e costante, la cui brevità comporta l'impossibilità di rispettarla allorche i cavi superano pochi metri di lunghezza. Ad esempio, allorchè il governo entra nello stato "attivo", inverando la linea ATN esso si aspetta che tutti gli apparecchi presenti scattino predisponendosi alla ricezione di comandi indirizzati o universali entro i due microsecondi.

Due metodi trasparenti d'estensione

Variano molto a seconda degli ambienti e del mezzo trasmissivo adottato. Su distanze relativamente ridotte (es. entro un edificio) le cose vanno in modo assai diverso, fin dalla fase di progettazione, rispetto al caso in cui si impieghino un modem ed un circuito televisivo. Principalmente può essere assai diverso il rapporto tra le velocità seriali: per estensioni fino a che centinaia di metri si possono adottare cavi coax fino ad 8 Mbit/sec. (o addirittura fibre ottiche) mentre coi modem full duplex necessari su linee di lunga distanza (telefoniche) si viaggia a 1200 bit/sec.
Nel primo caso, quello delle distanze brevi, l'estensione prevede che il Bus, in ogni "extender" sia connesso con una logica d'interfaccia munita di dispositivi specifici per controllare il transito dei byte. A tale scopo i dati paralleli vanno alternativamente da un'estremità all'altra dell'interfaccia dopo esser stati serializzati, in modo che ciascuna estremità sia aggiornata con lo stato logico della sua simmetrica. È in gioco un meccanismo di handshaking tale che la sequenza del trasferimento di un byte iniziata da un dispositivo-sorgente ad un estremo non possa concludersi su di esso, fino a quando l'altra estremità non ha a sua volta completato un trasferimento di byte.
Normalmente si usa un solo cavo, visto che le direzioni trasmissive si alternano. Il "frame" di dati seriali viene di solito completato con un breve segnale CRC (Cyclic Redundancy Check per ottenere, assieme all'isolamento metallico dei circuiti di pilotaggio del cavo, una pratica insensibilità al rumore elettrico, che solo nei casi di estremo inquinamento viene combattuta con sistemi più affidabili quali un doppio cavo in fibre ottiche.
L'estensione via modem comporta un più complesso protocollo di comunicazione a ben più lento ritmo, per cui gli extender su lunghe distanze sono attuate con circuiti microprocessorati. In particolare stavolta manca la correlazione rigida delle sequenze alle due estremità: ne deriva l'accumulo di molti byte in apposite memorie di transito prima della trasmissione. Molti byte vengono uniti in un "pacchetto" o in una "frame" dopo che insieme con ciascun byte di dati viene inviata una copia delle 5 linee di comando (IFC, ATN, SRQ, EOI e REN). Il che rende possibile la creazione di tali condizioni all'estremo remoto.
Accenniamo infine al fatto che i bus estesi rendono più seri i problemi della correzione degli errori, anche a causa dell'influenza dei fenomeni transitori. Gli extender seguono l'unico approccio possibile, la correzione tramite ritrasmissione dei pacchetti errati. A questi si aggiungono bit ridondanti di parità, longitudinali e trasversali per il controllo bidimensionale della parità in ricezione.

(Nota redatta su segnalazione della Hewlett Packard Italiana)

di **M. Morocutti**

Se uno psicologo sottoponesse un accanito lettore di BIT ad un test di libera associazione, proponendo la parola "microprocessore" si sentirebbe quasi certamente rispondere "personal computer". Si tratta infatti della più popolare applicazione del meraviglioso chip, e non ci si deve perciò meravigliare se per molti dire microprocessore equivale a dire Personal Computer.

Si possono però presentare molti casi in cui un microprocessore può cavarsela egregiamente, circondato solo dal minimo indispensabile. Anzi, talvolta impegnare un Personal per un certo lavoro può rivelarsi una soluzione eccessiva, sia dal punto di vista dell' impiego della macchina che da quello non trascurabile del costo del sistema. Se del computer ci interessa la parte di interfacciamento con l' esterno, mentre non ci serve il video e non si ha necessità di registrare dati su floppy disk, allora potremo prendere in considerazione l' idea di non impiegare un Personal, ma soltanto un semplice circuito a microprocessore. In questo caso parleremo propriamente di "microcomputer", che nel significato originario del termine sta ad indicare un "circuito costruito attorno ad un microprocessore". In altre parole, si tratta di costruire la parte principale di un hardware intelligente.

Quando si ha necessità di impiegare un hardware intelligente?

Forse non ci avete mai pensato, ma le risposte sono moltissime e sono limitate solo dalla vostra fantasia. Si possono ad esempio realizzare funzioni logiche molto complesse, che richiederebbero altrimenti una notevole quantità di integrati e che, soprattutto, sarebbero difficilmente modificabili.

Si possono costruire degli automatismi che controllano sequenzedi eventi, o pilotare dispositivi di una certa complessità.

Se avete la vocazione del fermodellismo, potrete automatizzare il vostro plastico ferroviario, dando così "intelligenza" ai vostri trenini. Se vi piace il rock potete ottenere sequenze luminose tipo discoteca, e se avete il pollice verde potrete realizzare un marchingegno vi che annaffi automaticamente il giardino, tenendo conto di orari, sensori di umidità e tipo di piante. Si può "dare l'anima" ad un robot, realizzare interfacce particolari o, se siete proprio fanatici, automatizzare l' intera casa. Tutto questo lasciando il vostro abituale computer libero per compiti che più gli si addicono, come ad esempio il calcolo, il word-processing o la grafica.

Requisiti indispensabili

Esaminiamo ora quali sono i requisiti di cui dovremo dotare il nostro microcomputer perchè svolga adeguatamente il suo compito.

Innanzitutto, bisogna pensare al microprocessore. Si deve scegliere un modello che sia diffuso, sufficientemente potente ma semplice da usare. Bisogna poi prevedere un certo numero di linee di I/O (cioè di Ingresso/Uscita), essenziali per collegare la nostra creatura con il mondo esterno.

Ci dovrà essere una certa quantità di RAM; ne basta poca perchè, a differenza di quella che troviamo in un Personal Computer, la nostra non dovrà contenere il programma, ma soltanto le variabili di funzionamento. Aggiungiamo perciò una EPROM per contenere il programma in modo permanente.

Non dobbiamo dimanticare che i componenti dovranno essere pochi e soprattutto reperibili con facilità, e che il circuito dovrà risultare compatto e poco costoso. Osserviamo che non è necessario avere a disposizione i bus del micro, perchè non c' è necessità di espandere la configurazione più di quanto ci siamo posti di realizzare.

Soluzioni adottate

La scelta è caduta su una CPU 6502. Questo microprocessore è molto diffuso grazie alla quantità di sistemi Apple che ci sono in circolazione, e grazie all' esistenza del SYM-1, dell AIM-65 e del leggendario KIM-1.

Per le funzioni di I/O si è adottata una VIA 6522, la Versatile Interface Adapter di cui si è già parlato sulle colonne di Bit e che è ben nota per essere usata sul SYM, sull' AIM e su qualche periferica collegabile all' Apple. Oltre che fornire 20 linee di ingresso/uscita, questo chip contiene due timer a 16 bit, che soddisfano le esigenze di chi necessita di temporizzazioni precise.

Un microprocessore a portata di mano

Il circuito stampato necessario alla realizzazione della scheda a microprocessore.

La RAM impiegata è la nota 2114, che contiene 1K x 4 bit. Adoperandone due si ottiene 1K x 8, una quantità di memoria sicuramente sufficiente per la maggior parte delle applicazioni.

Per contenere il programma si userà invece una EPROM del tipo a singola alimentazione, da 2 o 4K a seconda delle necessità. I modelli usabili sono la 2716 (Intel, SGS, National) o la 2516 Texas per i 2K, mentre potremo adoperare la 2532 Texas come memoria da 4K.

Per consentire un funzionamento preciso e ripetibile, il circuito monta un quarzo come elemento di temporizzazione del generatore di clock.

Il circuito, denominato Microcontroller, trovà posto su di uno stampato a doppia faccia con fori metallizzati. Per il montaggio dei circuiti integrati è certamente consigliabile adoperare degli zoccoli, evitando così problemi di saldatura e rendendo facile la sostituzione di eventuali esemplari difettosi o guasti, nonchè della EPROM che deve essere rimossa per cambiare il programma contenuto.

L' uso del circuito stampato consente di eliminare qualsiasi collegamento filato, che oltre a creare confusione, costituirebbe una possibile causa di malfunzionamento ed aumenterebbe le dimensioni della realizzazione.

Le connessioni con l' esterno sono realizzate tramite un connettore ad innesto diretto a 24+24 poli, con passo di 2.54 mm. Se il Microcontroller viene usato stabilmente in una ben definita applicazione, nessuno impedisce di saldare direttamente sullo stampato, sulle linguette del connettore, i fili che lo collegano con l' esterno. Non avendo a disposizione un connettore da 24+24 poli se ne può usare un altro di dimensioni maggiori (per esempio da 25+25), facendo ben attenzione ad inserire il circuito allineandolo con un bordo, e sempre con quello.

Bisogna tenere presente che, nonostante la cura posta nella realizzazione, questa scheda non è concepita per un uso pesante, come potrebbe essere quello in automatismi industriali. Mancano per questo alcuni requisiti, come il solder di protezione sullo stampato, zoccoli di qualità superiore, la doratura del connettore a pettine, ecc. Questo significa soltanto che il nostro circuito dovrà essere salvaguardato dalla polvere, dalle vibrazioni eccessive e dall' umidità, situazioni che potrebbero verificarsi in un ambiente industriale. Impiegato invece nelle nostre applicazioni "casalinghe" ed hobbystiche non mancherà di fare egregiamente il suo dovere senza inconvenienti.

Figura 3 - Lo schema elettrico della scheda a microprocessore.

Lo schema elettrico

Nella descrizione del circuito si suppone che chi legge abbia un minimo di domestichezza con l' argomento, che conosca cioè il significato di parole come data bus, address bus, clock, RAM, EPROM, e così via. Iniziamo ad osservare il circuito di decodifica. Le due linee di indirizzo più alte, cioè A14 ed A15, entrano in un decoder del tipo 2 to 4 (un 74LS139). Tale circuito ha quattro uscite, chiamate Y0, Y1, Y2, Y3. A condizione che l' ingresso di abilitazione G sia a livello zero, il decoder porrà a zero una delle sue quattro uscite a seconda della combinazione presente sugli ingressi A e B. In questo modo, per ogni coppia di possibili valori delle linee di indirizzo A14 e A15, il decoder selezionerà uno dei dispositivi presenti sulla scheda e collegati allo stesso data bus, cioè la RAM, la VIA o la EPROM. Un'uscita rimane inutilizzata.

La figura 4 riporta le zone di memoria che vengono di conseguenzaassegnate ai tre dispositivi; notate che ognuno di essi si ripeterà fino a riempire l'intero spazio a sua disposizione. Per capire perchè questo accada, consideriamo il caso della RAM. Trattandosi di un chip da 1K, ogni 2114 dispone di 10 linee di indirizzo, collegate ad A0...A9. Siccome il decoder agisce secondo A14 ed A15, gli indirizzi da A10 ad A13 non sono considerati, e quindi non contano. La nostra RAM si ripeterà perciò 16 volte nella zona di memoria che le è assegnata nella la fig. 4. Infatti, con i quattro indirizzi non collegati si possono fare sedici combinazioni. Per la VIA e la EPROM valgono discorsi analoghi.

Giunti a questo punto è bene fare una breve precisazione riguardo alla pronuncia dei numeri esadecimali. Trovandovi di fronte ad un numero esadecimale scritto 00, vi verrebbe spontaneo pronunciarlo "quattromila". Così facendo, però, si perde la distinzione tra il nostro 00 ed un vero 4000 (quattromila, questa volta). è bene perciò che il numero 00 si pronunci "quattro-zero-zero-zero", e così anche per gli altri numeri esadecimali. All' inizio è un pò noioso, ma ne vale la pena, credetemi.

Tornando ad esaminare lo schema di figura 3, notiamo che il bus dei dati è connesso a tutti i dispositivi tra i quali vi può essere scambio di dati. Altre linee che partono dalla CPU, come il Read/Write, sono invece connesse solo ad alcuni dispositivi.

Un segnale importante è il segnale di fase 2 ($\Phi 2$), che rappresenta il clock fondamentale dell' intero circuito, e in relazione al quale cambiano tutti gli altri segnali. Un esempio di questo discorso è il RAM R/W che viene generato dal

Un microprocessore a portata di mano

Figura 4 - La mappa di memoria della scheda Microcontroller.

74LS139 e portato alle RAM: esaminiamolo più da vicino. Nella figura 5 si possono vedere le relazioni tra vari segnali e la fase 2. Quando la CPU deve mettere un indirizzo sul bus degli indirizzi, lo fa poco dopo che il segnale di Φ2 è andato a zero. Il periodo in cui la Φ2 è a zero si chiama, guarda caso, fase uno. I dati invece vengono posti sul loro bus poco dopo l'inizio della fase due, ma il microprocessore manifesta l'intenzione di voler scrivere nella RAM già all'inizio della fase uno, tramite il segnale di R/W. Se questo segnale comandasse direttamente la scrittura nella memoria, quest'ultima immagazzinerebbe il dato in una locazione sbagliata, perchè proprio in quel momento gli indirizzi si stanno assestando (vedi figura 5). Occorre quindi dare il comando di scrittura alla memoria solo in fase due, quando gli indirizzi sono ormai sicuramente stabili. A questo provvede l'altra metà del decoder, e ve ne potete rendere conto considerando come cambiano le uscite al variare dei due ingressi, che in questo caso sono i segnali di R/W e di Φ2. L'uscita chiamata RAM R/W, cioè R/W per le RAM, va a zero solo quando il R/W è a zero e contemporaneamente la Φ2 vale uno.

Figura 5 - Fase 2 ed altri segnali.

La VIA 6522 non necessita di questo segnale, perchè lo ricava da sola internamente.

Un altro particolare da notare è che, per poter montare sia una EPROM da 2K che una da 4K, il piedino 18 dello zoccolo è collegabile a scelta tramite un ponticello (jumper). Collegandolo ad A11 si può montare una EPROM da 4K, mentre va collegato sull' altro lato per una EPROM da 2K.

Il clock ad 1 MHz è fornito direttamente dalla CPU, che sfrutta per questo un inverter del 74LS04 ed un quarzo. Il circuito adottato è quello indicato dal costruttore del 6502, che risulta essere molto semplice ma leggermente delicato: bisogna evitare di toccare i terminali del quarzo con le dita mentre è in funzione, perchè potrebbe mettersi ad oscillare stabilmente ad una frequenza di 3 MHz, impedendo il funzionamento del resto del circuito. Se succede questo, spegnendo e riaccendendo il Microcontroller tutto tornerà normale.

Una parte del 74LS04 è dedicata al circuito di RESET. Quando si da tensione all' insieme, occorre che l' ingresso di "RESET" della CPU sia mantenuto a livello basso fino a quando il circuito di clock *non si è avviato* correttamente, dando tempo anche alla tensione dell' alimentatore di stabilizzarsi. A questo provvede la rete di temporizzazione costituita dalla resistenza da 22K e dal condensatore da 10 uF.

Figura 6 - Disposizione dei componenti.

lato superiore		lato inferiore	
1	GROUND	1	
2	GROUND	2	
3	CA1	3	$\overline{\text{RESET}}$
4	PA0	4	
5	PA1	5	
6	PA2	6	
7	PA3	7	
8	PA4	8	
9	PA5	9	$\overline{\text{NMI}}$
10	PA6	10	
11	PA7	11	
12	PB0	12	
13	PB1	13	
14	PB2	14	
15	PB3	15	
16	PB4	16	
17	PB5	17	
18	PB6	18	
19	PB7	19	
20	CB1	20	
21	CB2	21	
22	CA2	22	
23	+5	23	
24	+5	24	

Figura 7 - I segnali sul connettore della scheda.

Collegando un pulsante tra la massa e l' ingresso di "RESET" del connettore (piedino 3 inferiore), si può resettare il circuito senza doverlo spegnere.

Una curiosità che emerge dallo schema è il modo in cui sono collegati gli indirizzi alla RAM: a chi ne conosce la piedinatura, essi appaiono di fatto "mescolati". Ciò è dovuto soltanto ad un fatto di convenienza nel disegnare il circuito stampato, *ma* non è difficile convincersi che questo collegamento non altera per nulla il modo in cui il microprocessore vede la RAM. Infatti, la sola cosa importante è che un dato sia rileggibile dalla stessa locazione nella quale è stato scritto, ma non in quale in assoluto si viene a trovare.

Il colore Ramtek rende più veloci le decisioni.

Ci sono infiniti modi di usare efficacemente la grafica a colori creata da un Ramtek 6211. Tanto per citarne qualcuno: CAD/CAM (Computer Aided Design), controllo visivo impianti di processo, sistemi di informazioni d'affari, cartografia, preparazione di mappe e relative quotature, ricerca scientifica.
Il 6211 è un terminale completo dotato di monitor ad alta risoluzione. Comprende un proprio linguaggio grafico a colori di facile uso che fornisce all'utilizzatore tutta l'interattività locale.
Si possono visualizzare 16 colori contemporaneamente da una tavolozza di 64 con una risoluzione di 640x512 punti indirizzabili.
Il 6211 dispone di una penna ottica e di tavoletta grafica.
Esiste anche in versione rack per costruttori di sistemi ed OEM.

STAMPANTE/PLOTTER 4100 COLORGRAPHIC

Il 6211 dispone poi di un'interfaccia standard per la Stampante/Plotter a colori 4100, in grado di produrre copie di alta qualità su carta normale, a basso costo. Il suo esclusivo sistema grafico a 4 testine produce copie a pieno colore in un unico passaggio. Senza problemi di toner esauriti o bagni chimici da cambiare. Ma usando solo nastri autorigeneranti in cartucce che si inseriscono velocemente in pochi secondi.
Il 6211 può anche essere usato con un sistema di riproduzione fotografico per ottenere diapositive a colori 35 mm, stampe di grande formato e lucidi a colori.
La piena compatibilità con il software applicativo Plot-10 e il VT 100™, rende ancora più semplice l'uso del colore agli utenti di questi sistemi. Di fatto il 6211 può lavorare con i più noti packages applicativi, compresi PATRAN™, DISSPLA™ e Gino-F™.

Per maggiori informazioni su questo terminale a basso costo ed alte prestazioni e sulla sua stampante, scrivete o telefonate direttamente alla nostra sede di Roma o Milano: saremo lieti di fornirvi tutta la documentazione che desiderate.

Sempre più società di software sono direttamente impegnate nello sviluppo e nella promozione di programmi applicativi per il 6211. Ecco le maggiori:

Rappresentazione grafica per uso statistico
ISSCO
SAS

Progettazione di Elementi Finiti
BYU
PDA
SDRC

General Purpose
Precision Visuals
GINO-F™

Desidero ricevere ulteriori informazioni sui nuovi sistemi Ramtek.
Gradirei:
☐ materiale illustrativo
☐ spiegazioni dettagliate da un vostro funzionario di vendita

Nome ____________
Cognome ____________
Qualifica ____________
Società ____________
Indirizzo ____________
Tel. ____________

Ramtek

Quando l'informatica dá spettacolo

Distribuita con tutta l'esperienza professionale della

Sede: Via V. Bellini, 24 - 00198 Roma - Tel. 06/867741 - Telex 613158 SAGARM **Filiale di Milano:** Via Plinio, 1 - 20129 Milano - Tel. 02/202761 - Telex 332677 SAGAMI

BLACK STAR

il sistema più...

intelligente, versatile, vantaggioso.

BLACK STAR è un sistema modulare che porta nel vostro ufficio la potenza e l'intelligenza dei grossi calcolatori, la versatilità dei modelli fatti per coprire ogni esigenza e i vantaggi che solo un sistema "made in Italy" può offrire.

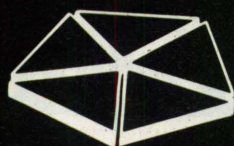
INTELLIGENTE. Progettato con soluzioni originali e innovative BLACK STAR è un microcomputer in grado di fornire prestazioni paragonabili a quelle di grossi calcolatori.

VERSATILE. La configurazione del sistema può essere adattata ad ogni esigenza grazie ai numerosi modelli, con capacità variabili da 800 KB a ben 96 M Byte.

VANTAGGIOSO. BLACK STAR offre tutti i vantaggi che solo un sistema "made in Italy" può offrire. Compresi i programmi sia di tipo commerciale che scientifico, studiati su misura per ogni applicazione.

Black Star
mod. 240 (2.4 MB)

Mini Black Star
mod. 80 (800 KB) - mod. 200 (2 MB)

Black Star
mod. 620 (6.2 MB) - mod. 870 (8.7 MB)
mod. 1120 (11.2 MB)
mod. 2120 (21.2 MB) - mod 120 x

pentasystem

PENTASYSTEM s.n.c. - 38100 TRENTO - Via Maccani 36, tel. 0461/981990

Realizzazione e collaudo

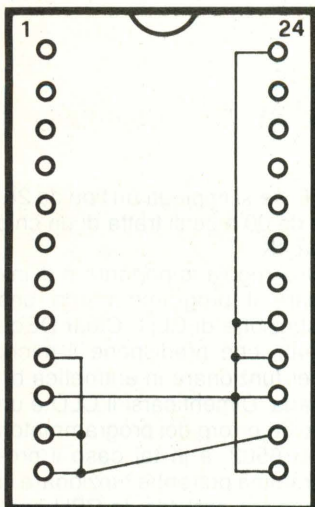
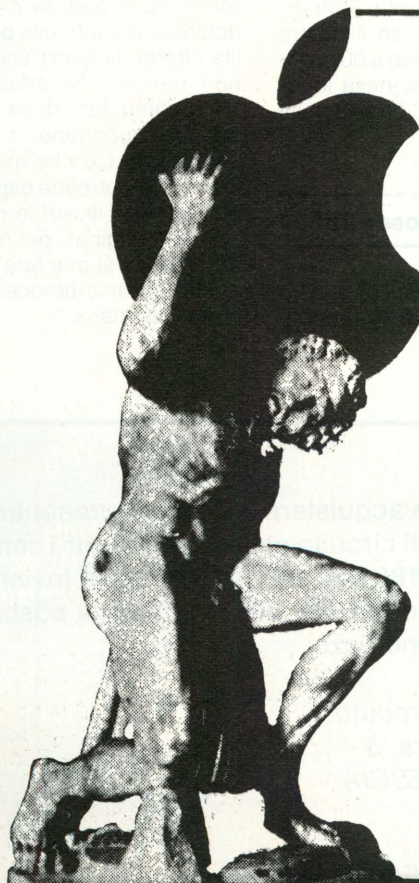
Non è difficile montare il Microcontroller sul circuito stampato, basta avere un saldatore a punta fine ed una mano ben ferma. Non *bisogna* assolutamente dimenticarsi di collegare il punto centrale del jumper 2K/4K ad uno dei due estremi. Riferirsi per questo alla figura 6.
È meglio montare prima le resistenze, i condensatori ed i diodi, poi gli zoccoli badando di infilarli bene a fondo nei fori dello stampato. Attenzione al verso di inserzione, riportato nella fig. 6. Attenti soprattutto alla polarità dei condensatori elettrolitici, che se collegato a rovescio possono esplodere, ed a quella dei diodi. Inserite poi le due 2114, i TTL, il microprocessore e la VIA sui rispettivi zoccoli. A questo punto saldate il quarzo, senza però ripiegarlo e incollarlo sulle RAM. È bene lasciare provvisoriamente i terminali lunghi, perchè sia possibile togliere le 2114 dagli zoccoli nel

N.B. i fili sono uniti solo nei punti indicati dal cerchietto nero

Figura 8 - Collegamenti per ottenere l'istruzione NOP.

caso sia necessario eseguire la procedura di ricerca guasti che è descritta più avanti. Quando si è sicuri del corretto funzionamento dell' insieme si piegheranno i terminali del quarzo fino a ripiegarlo sulle RAM, dove potrà essere incollato con il pezzetto di biadesivo nella posizione visibile in figura 6.

Figura 9 - I vettori del processore 6502.

Ora possiamo iniziare il collaudo. La prima cosa da fare è procurarsi un alimentatore da 5 volt in grado di erogare almeno 400 mA ed un oscilloscopio. Prima di connettere l' alimentazione è meglio dare un' attenta occhiata al lato inferiore dello stampato, per controllare che non si siano formati dei cortocircuiti accidentali durante la saldatura. Poi si può dare tensione, rispettando la polarità riportata in fig. 7 dove si vedono tutte le linee del connettore a 24+24 poli. Controllare con l' oscilloscopio la presenza di una quadra ad 1 MHz sul piedino 39 della CPU: è il segnale di clock Φ2.
Ci si deve rendere conto che non è possibile collaudare a fondo un circuito come il Microcontroller adoperando strumentazioni modeste; questo perchè la complessità interna dei componenti impiegati è notevole. Ciò significa che se ad esempio la RAM avesse un difetto di funzionamento, non ce ne accorgeremmo quasi certamente adoperando solo l' oscilloscopio: si dovrebbe usare per lo meno un programma di autotest, oppure togliere il componente sospetto e testarlo con qualche dispositivo creato apposta per quello scopo. Siccome però i guasti di un componente nuovo sono abbastanza rari, possiamo essere sicuri che la prima cosa a cui pensare in caso di malfunzionamento è un difetto di montaggio. In tal caso occorre controllare con estrema attenzione che duran-

te la saldatura non si siano accidentalmente cortocircuitati due punti vicini sullo stampato, o che non ci siano saldature fredde o addirittura dimenticate.
Controllare anche l' esatto verso di inserzione di tutti i componenti. Per la ricerca dei cortocircuiti può essere molto utile una lente di ingrandimento.
Se nonostante questo, una volta installato un programma il nostro circuito si rifiutasse di funzionare, c' è la possibilità di eseguire un test più approfondito, disponendo di un oscilloscopio e di uno zoccolo preparato appositamente.

Test dell'Address Bus e dei Chip Select

La prima cosa da fare è preparare uno zoccolo a 24 pin da inserire al posto della EPROM, al di sopra di quello già esistente. Tale zoccolo dovrà avere i pin collegati fra di loro in modo da formare il codice corrispondente all' istruzione di NOP (No Operation), cioè . I collegamenti si possono effettuare con spezzoni di filo rigido innestato nello zoccolo, seguendo il disegno della figura 8. Inserendo questa "finta EPROM", il microprocessore eseguirà continuamente l' istruzione NOP, cioè No Operation, e quindi gli indirizzi spazzoleranno continuamente l' intera area di memoria con regolarità. L' oscilloscopio ci sarà utile per controllare che ogni linea dell' address bus, da A0 ad A15, non sia in cortocircuito con qualcos' altro. Se va tutto bene, si devono vedere sulle linee di indirizzo una serie di onde quadre a frequenza via via decrescente. La linea A0 avrà la frequenza più alta, A15 la più bassa.
Mentre questo test è in funzione si possono controllare le uscite del decoder, che dovrebbero abilitare, nell' ordine, la RAM, la VIA, e la EPROM. Quando è selezionato uno di questi dispositivi non ne deve essere abilitato nessun altro, e tutti tre devono comunque essere selezionati per un quarto del periodo dell'onda quadra che è presente su A15.

Come programmare il microcontroller

L' argomento si divide in due parti, ovvero come adattare un programma al nostro circuito e cosa usare per svilupparlo e per inserirlo nella EPROM. Inizierò dalla seconda questione. Visto che si adopera il 6502, bisogna lavorare su un sistema che impieghi questo microprocessore. Il programma può essere agevolmente sviluppato e collaudato adoperando un AIM-65, oppure un SYM-1 od anche un KIM-1, meglio se servendosi di un assembler. Se possedete un Apple, nulla vieta di impiegare uno dei vari assembler disponibili.
Impiegando un piccolo sistema (SYM od altri) è più facile provare ciò che si sta mettendo a punto, in quanto si ha disponibilità di una VIA identica a quella che si trova sul Microcontroller. Con un Apple è già più problematico,a meno che non si disponga di una scheda aggiuntiva che monti una VIA con la quale provare se il programma fa il suo dovere, anche se è possibile farne a meno. Con un po' di abilità si può collaudare l' intero programma senza provare i collegamenti con l' esterno. Ancora, il programma potrebbe essere provato su un piccolo sistema ed essere poi scritto nella EPROM con un programmatore collegato ad un Apple. In ogni caso, la EPROM dovrà essere programmata.
Se possedete un programmatore collegato al vostro sistema il gioco è fatto, altrimenti bisognerà ricorrere all' aiuto di qualche volenteroso amico, o magari di qualche ancor più volenteroso venditore di computer.
Per quanto riguarda la prima questione, e cioè l' adattamento del programma al nostro microcomputer, bisognerà tenere conto di alcune regole essenziali. Prima di tutto gli indirizzi. La RAM inizia a 00 e finisce a F.F.. Non si deve usare la pagina uno, cioè da 00 a FF, se non con estrema attenzione, perchè contiene lo stack del microprocessore. La VIA ha indirizzo base 00, e si estende fino a 0F (sedici registri). Lo spazio occupato dalla EPROM dipende dal modello: andrà da 00 a FF, se si impiega un tipo da 2K, e da 00 a se si tratta di un chip da 4K.
Una regola importante è di iniziare il programma con una istruzione di CLD (Clear DEcimal), che predispone il micro per funzionare in aritmetica binaria. Dimenticarsi il CLD è un tipico errore dei programmatori del 6502, e in tal caso il programma potrebbe funzionare in maniera assurda: la CPU eseguirebbe le somme e le sottazioni in BCD invece che in codice binario.
Nelle primissime istruzioni, in pratica subito dopo il CLD, ci dovrà essere l' inizializzazione dello stack pointer, costituita da un LDX , seguito da un TXS. Si scriverà poi l' inizializzazione della VIA, che dipenderà ovviamente dal modo in cui si vogliono utilizzare le linee di I/O, i timer e le altre potenzialità del chip. Un' altra cosa da non dimenticare è il vettore di RESET, ed eventualmente quelli di IRQ e di NMI. Il vettore di RESET dovrà "puntare" all' inizio del programma e, secono la consuetudine del 6502, si scriverà prima la parte bassa dell' indirizzo e poi la parte alta. Riferendosi alla figura 9, se il nostro programma iniziasse a 00 si dovrebbe scrivere 00 nella locazione nella $FFFC ed $F8 nella $FFFD.

Per chi non conosce il 6502

Se non siete pratici di questo microprocessore, o se ritenete che la vostra conoscenza vada approfondita, potete leggere qualche libro sull' argomento. La Jackson ne ha pubblicati due che vanno benissimo:

- Programmazione del 6502
- Applicazioni del 6502

Quest' ultimo tratta anche dell'interfacciamento di un microcomputer con l' esterno, cosa molto importante per realizzare poi in pratica delle applicazioni. N‹ è escluso che si torni su questo tema con un successivo articolo. Vi sono anche indicazioni per utilizzare la VIA; se non lo si conosce bene, questo componente rischia di non essere sfruttato al meglio delle sue capacità, perciò è bene documentarsi.

Conclusioni

Ho descritto un circuito che, con poca spesa, permette di inserire un microprocessore nelle vostre realizzazioni. Esso può sostituire un Personal Computer in molti ruoli in cui non è richiesta per intero la potenzialità offerta da quest' ultimo. Ora non rimane che adoperare il Microcontroller dove più vi sembra opportuno, e magari segnalarmi ciò che avete realizzato. Bit potrebbe ospitare un articolo che illustri le realizzazioni più originali, per mostrare a tutti cosa si può fare quando si ha un microprocessore a portata di mano. ■

È possibile acquistare il kit del Microcontroller, completo di circuito stampato e di tutti i componenti, a L. 188.000 + IVA 18%. Basta inviare un assegno non trasferibile o un vaglia postale al seguente indirizzo:

Marco Morocutti
Via R. Serra, 5
25127 BRESCIA

La Nota

Personal Computer: un'applicazione del prossimo futuro

Parlando di PC, quasi sempre ci si scontra con il problema delle loro applicazioni.
Evidentemente questo non riguarda quanti hanno già ben presente l'utilizzo più o meno personale cui dedicare il PC, ma la grande massa dei potenziali utenti, spesso stimolati da messaggi pubblicitari sulle potenzialità applicative di cui non comprendono appieno il significato, anche perchè con ogni probabilità non vedono come e perchè dar seguito a quella data proposta applicativa.
Non parliamo poi delle varie occasioni di studio e dibattito sull'argomento, oramai all'ordine del giorno, sempre più con connotati socio-economici sulla futura società computerizzata, in cui tutti avremo l'inevitabile PC sul posto di lavoro, a scuola, a casa. Ma per fare cosa?.
La solita domanda a metà strada tra l'ozioso ed il provocatorio.
Una risposta ci viene dall'America, non tanto in termini di novità assoluta, ma piuttosto come risultato di uno sviluppo, di possibilità di organizzare una interessante situazione attorno al PC, senza dubbio esportabile anche in altri paesi, tra cui il nostro, posto si creino le condizioni, come dicevo, organizzative al riguardo.
La proposta applicativa viene da una società, la BRS/After Dark.
Di cosa si tratta?
Un primo sospetto al lettore dovrebbe venire dal nome stesso della società: After Dark, cioè dopo il crepuscolo, quindi alla sera, o meglio, anzi peggio, di notte.
Quindi, assonnati hobbisti che nel cuore della notte si dedicano a maniacali attività plagiati da inarrivabili miraggi di guadagno?
Per fortuna no, anche se di notte sempre si tratta.
Il discorso nasce dalla esistenza in USA di una diffusa rete di banche dati specializzate, cui è possibile collegarsi per avere le informazioni del caso.
L'operazione ha i suoi costi, in genere ragionevolmente sopportabili solo da un'azienda o comunque da persone che fanno uso professionale delle informazioni.
Il privato, il singolo, interessato magari ad ampliare le conoscenze in una data disciplina, quindi potenziale utente non professionale della banca dati, vede l'operazione come troppo costosa, non giustificabile. Ma come ben sappiamo, anche in Italia telefonate alla sera, fuori dalle ore calde, costa di meno.
Perchè non utilizzare allora il tempo delle Banche Dati, in cui le stesse sono praticamente inutilizzate, alla sera, di notte, quando gli uffici sono chiusi, la città dorme, ma non così l'indefesso studioso, il volenteroso nottambulo, perchè fornito di Personal Computer? Infatti l'unica condizione è il possesso di un PC, collegabile su linea telefonica alle Banche Dati disponibili.
Quali?
Alcuni esempi: chi si interessa di agricoltura può far uso del Data Base della National Agricoltural Library, per 8 dollari l'ora; chi invece pensa di avere un figlio "eccezionale", basta chieda il contatto, a soli 11 dollari l'ora, con il Council for Exceptional children.
Il pacchetto di proposte è in effetti molto ampio, in quanto la After Dark possiede a catalogo accessi a Data Base tra i più vari, da quelli centrati su informazioni di psicologia, a quelli di medicina, chimica, fisica, finanza, ed altri ancora.
La parola d'ordine è: Your Own University Library At Home (la vostra personale libreria universitaria a casa).
Come detto, questa proposta non è particolarmente eccezionale, prima di tutto perchè non è l'unica: non solo in USA ma anche in Europa vi sono iniziative analoghe.
Poi l'originalità è praticamente nulla, trattandosi di un fatto puramente organizzativo, nato sull'esperienza di interessanti banche dati e sulla verifica di una immensa diffusione di PC.
L'eccezionalità della cosa stà però, a mio parere nel momento di verifica reale di una corretta impostazione d'utilizzo dei PC, al di là di un fumoso discorso di potenzialità, più o meno vago.
A mio giudizio, parlando in termini più generali, il problema di fondo è quello di individuare una reale motivazione, anche limitata, per l'ingresso dei PC in casa, a scuola, in azienda.
Poi da lì espandersi verso qualsiasi campo o momento applicativo pensabile, una volta digerito la non indifferente modificazione del nostro comportamento, del nostro modo di organizzare le cose, imposta dalla presenza dei PC.
Quindi, ben vengano proposte come quella della After Dark, per ora solo americana, ma tra poco anche italiana, inevitabilmente.

Aldo Cavalcoli

di **M. Sarli**

Parte prima

Ada è il linguaggio di programmazione di cui più si sta parlando in giro negli ultimi tempi (**Bit** stesso ne ha accennato, al termine della serie sul Pascal). Commissionato dal Dipartimento della difesa Americano con lo scopo dichiarato di sostituire, senza alcun fallo per quanto riguarda le proprie applicazioni, tutti i linguaggi di programmazione finora usati, a causa dell'autorevolezza del committente (*il cui parco sistemi è, tra l'altro, sterminato.* Ndr) e della quantità di sforzi intellettuali ed economici profusi nello studio e nella ormai quasi completa realizzazione, *unita* ad una grande versatilità già prevista in sede di progetto, si presenta con tutte le carte in regola per incontrare un grosso successo.

Vediamo, in una rapida panoramica, le sue più importanti caratteristiche e la filosofia che ne è alla base.

```
procedura SOMMA is                    —— intestazione
    I,X : INTEGER;                    — parte dichiarativa: descrive i dati
    V:array (1...10) of INTEGER;
    begin                  —— parte esecutiva : contiene le istruzioni
        for I in 1 .. 10 loop
            GET(X);
            V(I):=X;
        end loop;
end SOMMA;      —— fine procedure SOMMA;
```

Figura 1 - Esempio di "procedure" in Ada

```
package COORDINATE is          —— intestazione del package specification
    type CART is        —— descrizione del tipo cartesiano
        record
            X : INTEGER;
            Y : INTEGER;
        end record;
    type POLAR is       —— descrizione del tipo polare
        record
            RO : FLOAT;
            TESTA : FLOAT;
        end record,
    function RENDI-POLAR (CORDCAR:CART) return POLAR;   —— elenco delle funzioni
    function RENDI—CART (CORDPOL:POLAR) return CART;   —— disponibili nel package
end COORDINATE; —— fine del package specification

package body COORDINATE is —— intestazione del corpo del package COORDINATE
    function RENDI—POLAR (CORDCART:CART) return POLAR is     —— implementazione
                                                    della —— funzione
                                                    RENDI—POLAR
        POLARI: POLAR;   —— variabile locale

        begin     —— inizio istruzioni
            POLARI.RO: = SQR (COORDINATE.X ★★2 + COORDINATE. Y★★2);
            POLARI.TETA:=arctg (COORDINATE.Y/COORDINATE.X);
            return POLARI;

    end RENDI—POLAR; —— fine implementazione RENDI—POLAR

    function RENDI—CART (CORDPOL (CORDPOL:POLAR) return CART is         —— implementa-
                                                              zione della ——
                                                              funzione
                                                              RENDI— CART

        CARTESIANE:CART;    —— variabile locale
        begin —— inizio istruzione
            CARTESIANE.X:=POLARI.RO★COS(TETA);
            CARTESIANE.Y:=POLARI.RO★SIN(TETA);
            return CARTESIANE;

    end RENDI.CART;   —— fine implementazione RENDI-CART
end COORDINATE; —— fine package body COORDINATE
```

Figura 2 - Le due parti costitutive di un package. Sopra si ha la specification e sotto il body (corpo) del package. Il secondo esprime modalità implementative (rese nascoste all'utente) dei compiti esplicitati nella specification.

Figura 5 - La parte specificativa (sopra) e (sotto) il corpo del package STACK serve ad illustrare come in Ada, vengono "incapsulati" i tipi astratti in modo tale che su di essi solo determinate operazioni siano lecite (cfr. testo).

Uno sguardo su Ada

ESPERANTO O BABELE NELL'INFORMATICA FUTURA?

Proporre addirittura la mitica Ada ai nostri lettori parrà a qualcuno esageroso, oltre che presuntuoso: oltretutto la Uno qui non c'entra nulla neanche alla lontana, visto che la creatura FIAT probabilmente si autodenomina così in quanto vuol essere, oltre che la numero uno, anche la prima di una serie (fortunata e non fortunosa si spera!). Ada invece, si sa, pretende di divenire non "un" linguaggio, ma "il" linguaggio. Per eccellenza ed autonomasia: vedi Ada e 'ppo mori, Ada e peu pü; per esprimere con due opposti dialetti le libidinosissime mire d'una nobildonna che esplicitamente ogni dialetto e lingua vuol soppiantare nell'EDP. Inutile dire che ambizioni così alte lasciano adito a parecchi scetticismi, ad onta della tremenda autorevolezza del suo sponsor, il ministero della difesa USA.
Ad un tal nientepopodimeno gli abituè che hanno il duro callo dell'incredulità informatica, senza lasciarsi troppo impressionare da sesteflotte e supermissili, prontamente obiettano argomenti che vanno dall'elencazione litaniosa delle lingue oggi esistenti (Sanctus Pascal: ora pronobis; FORTH venerabilis: miserere nobis....) alla citazione dei quasi bioritmici fallimenti cui sono andati incontro tentativi analoghi, a partire dal PL/1 l'unico oltretutto che, ponendosi come Ada lo scopo di fondere in sè i pregi ed i tool dei linguaggi orientati alla scienza o al business, ha finito, con la sua pletorieità, per esser usato in modo "dimidiato" solo da sparute frazioni d'utenti delle opposte sponde. Però adesso l'informatica tende sempre più all'integrazione di tutto e dappertutto, obietta timidamente un fan della Contessa di Lovelace, nata Byron.
Può darsi, incalzan subito i propugnatori della libera-lingua-in-libero-mercato, però, prima che venga fuori il sospirato (o sospiroso?) compilatore d'Ada, da un lato il patrimonio software accumulato in ambienti tradizionali (FORTRAN, COBOL, BASIC magari un po' di Pascal, più qualche linguaggio insostituibile in settori particolari) diverrà travolgente ancor più che oggi; d'altro canto chi sa che il settore dell'Intelligenza Artificiale non si decida a fare i promessi frutti quinto-generazionali e allora addio linguaggi, esperanti e volapuck inclusi.
Scendiamo più terra-terra, prova a far ragionare gli invasati un ammiratore di Ada: oggi che si hanno i micro a 16, presto a 32 bit Ada si diffonderà presto anche nell'EDP quasi-popolare. Anzi già su Byte compare la pubblicità di suoi microcompilatori e la Western Digital - quella che a suo tempo realizzò quella interessante macchina-Pascal denominata Microengine - si appresta a lanciare entro fino 83 un microsistema analogo per Ada, addirittura completo di tutti i complessi supporti dell'"ambiente" Ada... Ve li raccomando, interrompono i miscredenti, basicalisti (per lo più): così si ripeterà, anche per la pupilla del DoD (dopo esserla stata per Babbage & Lovelace) quel che è accaduto alle ambizioni esperantiche del Pascal che, per questa via, si è già disperso nei rivoli di frazionismo quasi pari a quello delle varie Internazionali.
Adesso però, signori miei, mi pare che esagerosi siate anche voi: in disfattismo, visto che osate dissacrare anche quel nobilissimo linguaggio di cui il guru dello strutturalismo informatico, Hoare ebbe a dire: *"Pascal è il migliore di tutti i suoi successori."* Ada escluso, of course.
Così tronco qui l'immaginaria-ma-non-troppo diatriba invitando la gente seria e culturalmente curiosa a leggersi questo paio d'articoli, godibili e chiari, per giunta. Solo così è possibile farsi un'idea precisa, anzichè perseverare nel pregiudizio.
Oltretutto, qualunque sia il destino futuro di Ada, specie nel mondo della microinformatica (che oggi come oggi appare indubbiamente piuttosto a sè stante...) è ragionevole attendersi che i nuovi concetti abbiano *comunque* pesanti ricadute in ogni dove.

(G.G.)

Un po' di storia

Lo sviluppo dell'uso dei calcolatori ha fatto crescere la complessità dei programmi il che ne ha drammaticamente aumentato il costo di sviluppo e di mantenimento. Partendo da queste considerazioni il DoD (Departement of Defense) decise di fissare l'attenzione sui settori che fino ad allora avevano proposto degli standard di linguaggio: il settore scientifico e quello industriale. Scopo dichiarato dell'iniziativa era di creare un linguaggio che soddisfacesse le necessità dell'Esercito, della Marina e dell'Aviazione e che allo stesso tempo desse luogo a costi di sviluppo inferiori. Fu così costituito nel 1975 un gruppo di lavoro, lo Higher-Order Language Working Group (HOLWG), col compito di occuparsi del problema.
Il primo passo fu quello di definire un insieme di specifiche cui il linguaggio doveva conformarsi. Esso avrebbe dovuto soddisfare sia applicazioni scientifiche che gestionali, consentire utilizzi efficaci in tempo reale, possedere un elevato livello di astrazione, non solo sulle strutture ma anche sui tipi di dati ed essere semplice da apprendere e da usare, pur conservando rigorosità e flessibilità. Queste e molte altre specifiche, anche non altrettanto precise, come "leggibilità" e "verificabilità", furono il risultato di una serie di rapporti sviluppati dallo HOLWG tra il 1975 ed il 1979 e chiamati in codice Strawman (1975), Woodman (1975), Tinman (1976), Ironman (1978) e Steelman (1979).
Il passo successivo fu quello di cercare tra i linguaggi di programmazione preesistenti il più vicino allo scopo, ma già all'epoca del rapporto Tinman uno studio esaustivo su 26 linguaggi, tra cui FORTRAN, COBOL, PL/1 Algol 60 e 68 Pascal, LISP, portò a concludere che al linguaggio ideale desiderato nessuno si conformava in modo soddisfacente.
I rapporti che seguirono il Tinman oltre ad allargare le specifiche, servirono a meglio definire le caratteristiche aggiuntive rispetto ai suoi predecessori. Esso infatti avrebbe dovuto essere sviluppato partendo da un linguaggio noto. Preliminarmente si scelsero Pascal, Algol 68 e PL/1.
Le specifiche furono pubblicate in un bando di concorso nel Maggio del 1977 e su 17 proposte di progetto ricevute ne furono scelte 4, ognuna delle quali fu denominata con un colore diverso. La corrispondenza colore-società fu mantenuta segreta, per evitare che le scelte fossero influenzate dal nome. Le 4 versioni furono chiamate Verde, Rossa, Gialla, Blu mentre le società scelte erano la CII - Honeywell Bull, la Intermetrics, la SRI International e la Softech.
Tutte scelsero come linguaggio di partenza il Pascal.
Le quattro versioni furono preparate in circa 8 mesi ed il progetto preliminare fu presentato nel Febbraio 1978 ad una commissione composta da rappresentanti del governo, docenti universitari ed esperti di problemi industriali. Dopo una prima selezione due versioni, la Verde e la Rossa, rimasero ancora in lizza. Furono completamente sviluppate per un anno ancora prima di essere definitivamente valutate da 15 team,, finchè nel Maggio del 1979 fu scelta come versione definitiva la verde (Green), proposta dalla CII-Honeywell Bull, ed elaborata da un gruppo di ricercatori francesi coordinati da Jean Ichbiah (*un franco-tunisino che, si direbbe rinverdisce le antiche glorie matematiche arabe. NdR*). Come è noto fu battezzata Ada, in onore di Ada di Lovelace, figlia del poeta Lord Byron

ed assistente di Babbage, l'inventore della prima macchina calcolatrice digitale programmabile della storia.

```
declare —— inizio block
      use COORDINATE;
                A: CART;   —— variabili locali del block
                B: POLAR;
      begin     —— inizio istruzioni
                     A.X:=5;
                     A.Y:=10;
                     B:=RENDI-POLAR (A);
end          —— fine del block
```

Figura 3 - Esempio d'uso del package COORDINATE *di cui la specification e il body sono riportati in figura 2.*

```
package TEMPO is

          type GIORNI is (LUN, MAR, MER, GIO, VEN, SAB,
          DOM); —— tipo enumerativo
          type DATA is 1..31; —— tipo subrange
          type MESI is (GEN, FEB, MAR, APR, MAG, GIU, LUG,
          AGO, SET, OTT, NOV, DIC);

          X : GIORNI; —— variabile di tipo GIORNI
          Y : DATA;   —— variabile di tipo DATA
          Z : MESI;   —— variabile di tipo MESI
end TEMPO;

procedure ESEMPIO is
       use TEMPO   —— in questa procedura si
                   —— usa il package tempo
       begin
              X:= LUN;
              Y:=28;
              z;=MAG;
end;   —— fine procedure
```

Figura 4 - Sopra è riportato un esempio di package di soli oggetti. In basso la procedure ESEMPIO *costituisce un caso applicativo del package* TEMPO.

```
package STACK is
type STACK is private

   procedure PUSH (I:in INTEGER; S:in out STACK); —— elenco delle funzioni
   procedure POP (I:out INTEGER; S:in out STACK); —— e procedure con cui
   function VUOTO (S:in STACK) return BOOLEAN; —— si può manipolare il
   function PIENO (S:in STACK) return BOOLEAN; —— tipo STACK

   private —— questa parte è invisibile all'utente

       type STACK is
                         record
                              TOP:INTEGER: = 0; —— numero elementi nello stack. È
                                                —— inizializzato a 0
                   PROF:array(1..100) of integer;      ——costituzione dello stack
                   end record;

end STACK

package body STACK is
     procedure PUSH (I:in INTEGER; S:in out STACK);

     begin   —— inizio implementazione di PUSH

             if S.TOP<100               —— se lo stack ha post liberi
                  then S.TOP:= S.TOP+1;     —— inserisci il nuovo elemento
                       S.PROF(S.TOP):=I;
             else PIENO (S); —— altrimenti chiama la function PIENO
          end if; —— fine della if

     end PUSH; —— fine della procedura PUSH

     procedure POP (I:out INTEGER; S:in out STACK);

     begin  —— inizio implementazione di POP

             if S.TOP>0           —— se ci sono elementi nello stack
                then I:=S.PROF (S.TOP); —— estrai il primo elemento
                     S.TOP:=S.TOP — 1;
                else VUOTO (S);   —— altrimenti chiama la function VUOTO
             end if;  —— fine della if

     end POP; —— fine della procedure POP

     function PIENO (S:in STACK) return BOOLEAN is

        begin —— inizio function PIENO
              return S.TOP=100;
        end PIENO;   —— fine function PIENO

     function VUOTO (S:in STACK) return BOOLEAN is

        begin —— inizio function VUOTO
              return S.TOP=0;
     end VUOTO; —— fine function VUOTO
end STACKS —— fine del package body STACK
```

Una versione preliminare del linguaggio standard fu pubblicata nel Giugno 1979 ed una sua revisione nell'Aprile 1980. Il compilatore accettato dal Dipartimento della Difesa non è stato ancora terminato.

La modularità del linguaggio

Una delle caratteristiche salienti di Ada è la possibilità di dividere grossi programmi in più parti che, a differenza di quanto avviene in Pascal, possono essere, sviluppate e compilate separatamente. Chiameremo queste parti unità di compilazione. In Ada esistono quattro tipi di tali unità: le procedure, le function, i package ed i task. Le procedure e la function hanno la stessa natura delle omonime del FORTRAN o del Pascal e constano di una parte dichiarativa ed una parte esecutiva. Nella figura 1 è presentata una semplice procedura che riceve dall'esterno 10 interni (istruzione GET) e li pone in un vettore. Per inciso, tutto ciò che è preceduto da due lineette "——" costituisce un commento.

Ben più importanti e innovative sono le unità di compilazione chiamate "package". Questi moduli possono essere usati per definire un insieme di risorse collegate logicamente. Un package è diviso in due parti, che a loro volta possono essere compilate separatamente: il "package specification" ed il "package body".

Il package specification rap-

presenta la parte dichiarativa che informa l'utente su tutte le risorse costitutive di quel package, il che ne rende possibile un utilizzo trasparente. Questa parte del package va compilata con l'unità di compilazione che lo usa.
Il package body, il "corpo" del package, esprime le modalità esecutive dei compiti che il package deve svolgere. Questa parte è nascosta all'utente e può essere compilata separatamente.
In sostanza il package specification dice "cosa" fa il package mentre il package body contiene il "come" viene fatto. L'utente conosce solo cosa può fare con quel package.
In figura 2 a titolo d'esempio, vi è in alto la parte dichiarativa di un package chiamato *COORDINATE*. In esso vi è anzitutto la dichiarazione di due nuovi tipi. Il primo *CART*, è un record con due campi interni che contiene le coordinate cartesiane di un punto nel piano.
Il secondo, *POLAR*, è anch'esso un record, ma ha due campi reali (in Ada detti *FLOAT*), e contiene le coordinate polari di un punto. Seguono, nel package specification, le due funzioni che vengono messe a disposizione, *RENDI CART* e *RENDI POLAR*. Quest'ultima prende in ingresso le coordinate cartesiane di un punto e restituisce (si noti la clausola "return") un tipo *POLAR* che contiene le coordinate polari dello stesso punto. La conversione inversa viene invece fatta dalla funzione *RENDI-CART*.
L'implementazione delle funzioni è riportata nel package body *COORDINATE* nella figura 2, sotto.
Come si nota tutto ciò che l'utilizzatore del package deve conoscere per usarlo è contenuta nella parte dichiarativa, che, ripetiamo, dovrà essere dichiarata e compilata insieme all'unità di compilazione che ne fa uso. In figura 3 vi è un esempio di utilizzo. Incidentalmente essa illustra anche un'altra caratteristica di Ada: la parte compresa tra "declare" ed "end" rappresenta un block (blocco). Si tratta di un modulo anonimo che può essere posto così com'è in qualsiasi parte esecutiva di una unità di compilazione. Si noti la clausola "use" che mette a disposizione del block le risorse del package *COORDINATE*.
Abbiamo visto cos'è un package e come si usa. Qualcuno però si chiederà quali sono le principali ragioni di questa sua duplice costituzione. Nello sviluppo di grossi programmi è molto utile la suddivisione in blocchi di minore complessità e funzionalmente ben definiti. In Ada una volta effettuata la suddivisione è sufficiente specificare, nelle unità di compilazione, solo le eventuali parti dichiarative dei package usati. L'intero programma potrà così essere compilato anche se non sono ancora presenti tutti i relativi package body. Naturalmente non si potranno assoggettare a test quelle parti che chiamano in causa i package "in sospeso" almeno fino al momento in cui i relativi body non saranno presenti. I package body potranno essere sviluppati in seguito oppure parallelamente, provati e compilati separatamente. Una volta pronto tutto il programma, con tutti package body al completo, lo si potrà collaudare nella sua interezza. Se qualcosa non dovesse andare per il suo verso, basterà (e sarà anche più facile) individuare i package body responsabili, modificarli e ricompilarli, senza dover ricompilare tutto quanto.
È importante notare come la parte dichiarativa dei package funga da interfaccia tra il programma che lo usa ed il package body. Quest'ultimo potrà poi attuare i suoi compiti in un modo qualsiasi, purchè esatto, e come lo faccia non ha nessuna importanza per l'utilizzatore, che può pertanto non conoscerne il listato. Questa caratteristica permette la creazione di *software-bus* i quali consentono un più duraturo impiego dello stesso programma principale, oltre ad un reimpiego in altri programmi degli stessi moduli, con evidenti tagli nei tempi di sviluppo.
Un'altra caratteristica dei package è la possibilità di contenere solo "oggetti", cioè un insieme di tipi e variabili. In figura 4, in alto, se ne ha un esempio e, in basso, un caso applicativo. Si noti la clausola "use" nella parte dichiarativa della procedura *ESEMPIO*, e l'assenza in quest'ultima di qualsiasi altra dichiarazione, dovuta al fatto che la parte esecutiva, compresa tra "begin" ed "end", userà tipi e variabili del package *TEMPO*.

I tipi astratti

Si è già detto dell'importanza della modificabilità di un programma, affinchè possa adattarsi flessibilmente a sopraggiunte esigenze, oppure essere reimpiegato per compiti moderatamente diversi da quelli di progetto. La modularità di Ada, attraverso i package, permette questo, ma in più isola i moduli dal resto del programma e rende inaccessibili le informazioni che riguardano come il corpo del modulo esegue la sua funzione.
Quanto detto è esteso in Ada non solo alle strutture di controllo ma anche ai tipi. Per molto tempo infatti si è adottata una stesura dei programmi orientata alle strutture di controllo. Il punto centrale veniva ad essere così la descrizione degli algoritmi, mentre la descrizione dei dati sui quali si operava restava marginale. Da diverso tempo, se ne è cominciato a parlare verso la metà degli anni settanta, si dà sempre maggior rilievo alla scrittura di programmi orientata ai dati, che sono divenuti così il nocciolo centrale, relegando gli algoritmi ad un ruolo non certo secondario, però di entità ausiliarie che indicano le relazioni e le operazioni che si compaiono su di essi. È quanto viene fatto nella definizione di tipi di dati detti *tipi astratti*. Si tratta di un nuovo tipo, espresso in funzione di altri più elementari cui vengono associate operazioni che sono le uniche tramite le quali si può manipolarlo. Anche qui, rimane inaccessibile all'utente il modo in cui ciò sia realizzato in concreto, in quanto quelle che a lui interessano sono le proprietà e le operazioni che il tipo possiede.

Uno sguardo su Ada

In Ada i tipi astratti vengono *incapsulati* in package.
Per meglio chiarire la natura di un tipo astratto partiamo da un esempio molto conosciuto. Lo stack o catasta e, come dice il nome, una pila di elementi, tipicamente di dati, il cui primo elemento, quello in cima, è l'ultimo aggiunto. Le operazioni che si fanno, in genere, su uno stack sono due: la *push*, che consiste nel porre un elemento sulla sua sommità, e la pop, che si compie prelevando l'elemento dalla cima della pila. Vediamo come in Ada si può implementare un tipo astratto stack. L'incapsulamento si effettua in un package e nella figura 5, sopra, ve n'è uno che descrive il tipo astratto *STACK*. La clausola *private* specifica che questo tipo può essere manipolato solo con le procedure e le funzioni indicate. Le procedure *PUSH* e *POP* svolgono i compiti già detti secondo modi che dovrebbero presentarsi autoesplicativi, mentre la funzione *PIENO* diventa vera quando lo stack è completo (overlow) la *VUOTO* quando è in condizioni di underflow. Il modo in cui lo stack è implementato, descritto dopo la clausola private, come si vede è un array di interi. La parte che segue private come vedremo non è visibile all'utente. Infatti in Ada vi è una distinzione tra l'interfaccia logica di un package e l'interfaccia fisica dello stesso. L'interfaccia logica consta di tutto ciò che è visibile all'utilizzatore e comprende quello che gli necessita per usare il package. In figura 5 è la parte compresa tra i termini "package" e "private" della package specification, mentre la rimanente, che segue "private" e termina con "end" non viene vista dall'utente e rappresenta, insieme alla precedente l'interfaccia logica, che viene passata al compilatore. Questo perchè mentre l'utilizzatore può non conoscere la struttura fisica del tipo astratto, tale conoscenza è necessaria per il compilatore. Di passaggio facciamo notare che nei parametri delle procedure si distinguono quelli che possono essere solo letti (in), parametri che possono essere oggetto di sola scrittura (out) e parametri che possono essere sia letti che scritti (in out).

Il package body *STACK* è nella stessa figura, in basso. Per inciso va detto che nel package body possono essere implementate procedure e funzioni, oltre a variabili, che non sono visibili nelle package specification, ma che servono ad implementare le procedure e le funzioni visibili.
Quando conviene l'approccio "data oriented"? Ciò si verifica in particolare nei sistemi che operano in tempo reale, ove le strutture dei dati rappresentano lo stato del sistema che evolve nel tempo, ma in generale in tutte quelle applicazioni in cui i dati e la loro manipolazione formano il cuore del problema, caso questo che si verifica con particolare frequenza nelle applicazioni gestionali. Nei problemi scientifici in genere la complessità del calcolo è di gran lunga superiore alla complessità dei dati, e dunque conviene di solito l'approccio tradizionale orientato agli algoritmi. Non a caso il FORTRAN, nato in ambiente scientifico, permette solo quest'ultimo tipo di approccio. La cosa non è cioè solo da porre in relazione a fattori storici, ci sembra.

(continua)

Bit

Reporter:

stampa delle variabili di un programma Applesoft

INTRODUZIONE

REPORTER, che è il programma descritto in questo articolo permette la stampa o la visualizzazione dei valori di tutte le variabili o gli elementi di array attivati da un altro programma Applesoft, e costituisce quindi un indispensabile strumento per il debugging e lo sviluppo dei programmi; è infatti possibile, ad esempio, trovare rapidamente variabili che assumono valori esterni al range consentito, oppure localizzare elementi di array non inizializzati. Questo programma è inoltre utile per la documentazione di programmi BASIC, in quanto fornisce una "cross reference table" delle variabili usate, accanto alle quali l'utilizzatore può scrivere il significato delle variabili stesse. Esso è previsto per l'uso su un sistema con almeno un disk drive, ed è completamente trasparente per l'utente, in quanto dopo aver operato ricarica da disco l'intero stato dell'Apple, salvato al momento della partenza del programma.

Struttura del programma

Affinchè sia possibile mantenere l'intero stato dell'Apple, e per non perdere il valore delle variabili al momento del caricamento del programma, esso deve essere fatto partire tramite un file, con un comando EXEC. Il file REPORTER provvede immediatamente a salvare su disco i puntatori, le variabili numeriche, le stringhe e il programma, per poi lanciare un programma BASIC (REPORTER.P) che calcola e stampa i valori delle variabili. Infine viene chiamato con un comando EXEC, dal programma stesso, un altro file che ricarica i dati ed il programma precedentemente salvati, per ripristinare interamente lo stato dell'Apple.

Mappa di memoria dell'Apple e codifica delle variabili

I programmi BASIC sono memorizzati, nell'Apple, a partire dalla locazione di memoria $0801 (2049 in decimale). Di seguito ad esso si trova la zona riservata alle variabili semplici, che contiene i valori delle variabili intere e reali, e dei pointer, uno per ogni variabile stringa, che puntano all'indirizzo di memoria a partire dal quale è memorizzata la stringa. Dopo questa zona se ne trova una analoga, che contiene i valori degli array interi e reali, ed i puntatori degli array stringa. Scendendo dall'alto della memoria, che in un sistema 48K con il DOS installato si trova a $9000 (36864), vi è la zona dove sono memorizzate le stringhe, detta stringpool. Tra l'inizio dello stringpool e la fine della zona riservata agli array vi è memoria libera.

Ovviamente è necessario all'interprete BASIC sapere dove sono i "confini" tra queste zone, ed essi sono memorizzati in pagina zero, come dalla tavola seguente, riportata dal manuale Applesoft:

$69-$6A: Puntatore all'inizio delle variabili semplici.
$6B-$6C: Puntatore all'inizio degli array.
$6D-$6E: Puntatore alla fine degli array.
$6F-$70: Puntatore all'inizio dello stringpool.
$73-$74: Puntatore alla fine dello stringpool.

Non tutte le stringhe vengono però memorizzate all'interno dello stringpool: infatti se in un programma è contenuta una istruzione del tipo

A$= "STRINGA DICHIARATA NEL PROGRAMMA"

il puntatore della variabile A$ viene posto ad indicare il punto di inizio della stringa all'interno del programma, ed essa non viene copiata nello stringpool, per risparmiare memoria. È quindi necessario avere in memoria anche il programma per sapere il valore di tutte le stringhe da esso attivate.

Poichè l'uso di un programma BASIC o di variabili in modo diretto altera il valore di questi puntatori è necessario, salvare su disco questi puntatori prima di lanciare il programma; questo è ciò che fa per prima cosa il file REPORTER, (fig. 1). Questo file successivamente salva come file binario il programma, le variabili semplici e gli array, e lo stringpool, per poi far partire il programma vero e proprio.

Descrizione del programma

Il programma REPORTER.P dapprima suddivide la memoria in una zona destinata al programma stesso e alle sue variabili, ed in una zona ove ricaricare da disco i dati salvati in precedenza, ponendo HIMEM=9987.

Successivamente il programma carica a partire dalla locazione di memoria 9988, primo byte libero, i puntatori salvati in precedenza dal *file REPORTER* con il nome PUNTA. Avendo ora in memoria questi puntatori è possibile calcolare gli indirizzi dei "confini" tra le zone in cui il programma in sviluppo aveva diviso la memoria. Dati questi valori è possibile calcolare innanzi tutto se vi è spazio sufficiente in memoria per le variabili da analizzare; infatti su un sistema 48K abbiamo ancora circa 26000 byte liberi (da 10000 a 36864) è necessario quindi calcolare l'ingombro della parte numerica delle variabili, che si ottiene sottraendo da EN (indirizzo di fine delle variabili) SV (indirizzo di inizio delle variabili), più l'ingombro dello stringpool, che vale SS (inizio dello stringpool) meno ES (indirizzo di fine della memoria libera), e verificare quindi che la somma di questi valori sia superiore a 26000. Nel caso non vi sia spazio sufficiente non è possibile avere la stampa delle variabili; viene quindi stampato il messaggio "NON C'È SPAZIO SUFFICIENTE" e viene ricaricato il programma precedente, con le sue variabili, chiamando il file REPORTER.E. Altrimenti l'esecuzione del programma continua con il caricamento da disco della parte numerica delle variabili, a partire dall'indirizzo 10000, e dello stringpool, che viene posto consecutivamente nella memoria. La mappa di memoria del sistema dopo questa serie di operazioni viene rappresentata in figura 3.

Nelle righe tra la 150 e la 200 vengono calcolati i nuovi valori degli *indirizzi di inizio* e fine delle varie zone di memoria, e vengono calcolati tre offset: D1 dice di quanto è stato spostato l'inizio delle variabili; D2 indica lo spostamento dell'inizio della zona riservata alle stringhe, e D3 di quanto deve essere spostato l'inizio

del programma. Infatti, se lo spazio in memoria è sufficiente, il programma da analizzare viene caricato dopo lo stringpool; D3 viene anche utilizzata come flag per indicare se il programma è stato caricato o meno: nel caso non vi fosse spazio per il programma D3 vale zero.
La zona di programma dalla riga 200 alla 235 serve a chiedere all'utilizzatore quali variabili vuole vedere, e se vuole visualizzarle sul video oppure ottenere un listato sulla stampante. Viene inoltre richiesto se si vuole la stampa degli elementi nulli degli array numerici, mentre quelli degli array alfanumerici non vengono mai stampati. Viene anche chiesto se si vuole la stampa di un avvertimento per le stringhe (variabili semplici ed array) i cui puntatori indicano una zona all'interno del programma. Questa caratteristica si rivela di estrema utilità per programmi che utilizzano la routine CHAIN descritta nel manuale del DOS, che permette di passare le variabili da un programma Applesoft all'altro; questa routine richiede infatti che non vi siano variabili stringa all'interno del programma; è utile ricordare in proposito che anche le stringhe dichiarate con la coppia di istruzioni READ - DATA non vengono copiate nello stringpool, e che dopo l'esecuzione di un programma del tipo di quello in fig. 4 i pointer della variabile A$ punteranno allo stringpool, ma quelli di B$ punteranno ancora all'interno del programma.
Nel ciclo tra la riga 240 e la 400 viene effettualta la stampa delle variabili semplici: per prima cosa la variabile indice I viene posta uguale ad SV, che indica l'indirizzo di inizio delle variabili semplici. Nel caso *non* sia richiesta la stampa di alcun tipo di variabili semplici, alla riga 245 l'indice I viene posto all'inizio della zona ove sono memorizzati gli array (SA), e il ciclo di stampa delle variabili semplici viene saltato. Alla riga 260 ha inizio il ciclo vero e proprio, con la chiamata della subroutine 5000, che trova il nome ed il tipo della variabile il cui indirizzo in memoria è puntato da I. Il nome delle variabili è infatti codificato in memoria in due byte che contengono il codice ASCII del nome della variabile, e, nel bit 7 di entrambi i byte, l'indicazione del tipo di variabile. Questa subroutine restituisce quindi in B$ il nome della variabile, e in FL un numero che vale 0 in caso di variabile reale, 2 in caso di variabile stringa e 3 per le variabili intere. In dipendenza dal valore di FL viene chiamata la subroutine che stampa il valore delle variabili del tipo appropriato:La subroutine 2000 chiama la subroutine 20000, che calcola il valore della variabile che si trova in memoria a partire dalla locazione puntata dalla variabile I. e da il risultato in T. Successivamente la 2000 stampa il nome ed il valore appena calcolato; poichè il valore delle variabili reali (escluso il nome) occupa sempre cinque byte in memoria, sia che siano semplici, come in questo caso, oppure elementi di array, il pointer I viene incrementato di 5 dalla subroutine 20000, in modo da puntare direttamente al nome della variabile successiva.
La routine che ha inizio alla linea 3000, riguardante le variabili stringa, dapprima preleva dalla memoria la lunghezza (L), e calcola in ST l'indirizzo di inizio della stringa. Se uno di questi valori è zero la stringa è stata dichiarata nulla, e quindi non viene stampata; altrimenti si chiama la routine 30000, che stampa la stringa, con eventuali note nel caso il valore di ST sia esterno allo stringpool, oppure con la stampa dei codici ASCII della stringa, nel caso vi siano dei caratteri con codice superiore a 127.
Alla riga 4000 ha inizio una routine analoga alla 2000, che stampa le variabili semplici di tipo intero.
Alla riga 450 inizia la zona di programma che stampa gli array; la subroutine 5000 serve nuovamente per sapere il nome e il tipo dell'array, che viene stampato immediatamente.
Alla riga 520 viene assegnato ad M il valore dell'indirizzo di inizio dell'array successivo, e nella linea 530 si legge il numero di dimensioni dell'array.
Il ciclo FOR - NEXT seguente carica nel vettore A% il valore di ogni dimensione dell'array in analisi, che viene poi stampato tra parentesi quadre nella riga 560.
La 620 stabilisce se deve essere fatta la stampa degli elementi dell'array. In caso affermativo vengono stampate le coordinate dell'elemento, e, chiamando la subroutine appropriata al tipo dell'array, si stampa il valore dell'elemento.
Nelle righe seguenti il programma calcola le coordinate dell'elemento successivo, e nel caso abbia finito di stampare l'array, passa al successivo, trasferendo il valore M calcolato in precedenza ad I. Se questo indice ha finalmente raggiunto il valore di EN, la stampa degli array è finita ed il programma chiama il file REPORTER.E, che ricarica il programma originale.
Le subroutine 12000, 13000 e 14000 sono analoghe rispettivamente alle 2000, 3000 e 4000, ma provvedono alla stampa di un elemento di un array, che differisce generalmente da una variabile semplice per lo spazio occupato in memoria.

File ausiliari

I due file REPORTER e REPORTER.E, che servono rispettivamente a lanciare ed a terminare l'esecuzione di questa utility, sono riportati nelle figg. 1 e 5. Per facilitare la creazione di questi file, che contengono un notevole numero di caratteri non registrabili direttamente sul disco, come le virgolette, sono riportati in fig. 6 e 7 anche il listati di due programmi, CREAREPORTER e CREAREPORTER.E, che registrano su disco i due file.

Installazione del programma

Per poter finalmente utilizzare questo programma bisogna quindi, oltre a copiare il programma riportato in fig. 2 e registrarlo sul disco con il nome REPORTER.P, scrivere i due programmini CREAREPORTER e CREAREPORTER.E, ed eseguirli. Dopo questo si può iniziare lo sviluppo di un programma, ed in qualsiasi momento ottenere la stampa delle variabili con il comando EXEC REPORTER.
Nel caso il programma non possa stampare le variabili poichè non è sufficiente lo spazio in memoria è opportuno ricordare che è possibile aumentare la memoria

```
FILE REPORTER

?CHR$(4)"BSAVEPUNTA,A105,L12
?CHR$(4)"BSAVENUMERI,A"PEEK(106)*256+PEEK(105)",L"(PEEK(110)*256+PEEK(109)-PEEK(
106)*256-PEEK(105))
?CHR$(4)"BSAVESTRINGHE,A"PEEK(112)*256+PEEK(111)",L"(PEEK(116)*256+PEEK(115)-PEE
K(112)*256-PEEK(111))
?CHR$(4)"BSAVEPROGRAMMA ,A1024,L" PEEK (106) * 256 +  PEEK (105) - 1024
RUNREPORTER.P
```

Figura 1

disponibile con una istruzione ?FRE(0); questa istruzione libera parte dello string-pool, diminuendo il suo ingombro in memoria, e ciò può bastare per avere spazio anche per il programma REPORTER.P con le sue variabili.
Se il programma deve essere usato su un sistema con meno di 48K è necessario rivedere la costante 26000 alla riga 150, e porla uguale alla quantità di memoria libera effettivamente disponibile sul sistema, meno i circa 8K necessari per questo programma, e lo spazio per il DOS. Nel caso si renda necessario un ulteriore risparmio di memoria, è possibile spostare l'HIMEM del programma REPORTER verso il basso, e spostare di conseguenza l'indirizzo a cui vengono caricati i puntatori delle variabili. Questo arreca però l'inconveniente di rallentare l'esecuzione del programma, in quanto si rende necessaria più spesso l'esecuzione automatica del riordino delle variabili, che è una operazione abbastanza lunga.

```
10  HIMEM: 9987
15 D$ =  CHR$ (4)
20  PRINT D$"BLOADPUNTA, A9988"
90  DIM C$(3): C$(2) = "$": C$(3) = "%"
100 SV =  PEEK (9989) * 256 +  PEEK (99
88):  REM LOMEM
110 SA =  PEEK (9991) * 256 +  PEEK (99
90):  REM INIZIO DEGLI ARRAY
120 EN =  PEEK (9993) * 256 +  PEEK (99
92):  REM FINE DELLA ZONA NUMERICA
130 SS =  PEEK (9995) * 256 +  PEEK (99
94):  REM INIZIO DELLE STRINGHE
140 ES =  PEEK (9999) * 256 +  PEEK (99
98) - 1:  REM HIMEM
144  :
145  REM  INIZIALIZZAZIONE E CARICAMENT
O DELLE VARIABILI
146  :
150  IF EN - SV + ES - SS > 26000 THEN
 PRINT "NON C'E' SPAZIO SUFFICIENTE": G
OTO 800
160 D1 = 10000 - SV:  PRINT D$"BLOADNUME
RI, A10000"
170 SV = 10000: SA = SA + D1: EN = EN + D
1
180 D2 = EN + 1 - SS:  PRINT D$"BLOADSTR
INGHE, A"EN + 1
190 SS = EN + 1: ES = ES + D2
192 D3 = ES - 1024
193  IF EN - 1024 + ES - SS > 26000 THE
N  PRINT "NON C'E' SPAZIO PER LE VARIAB
ILI STRINGAALL'INTERNO DEL PROGRAMMA": S
7$ = "S": D3 = 0
195  IF D3 THEN  PRINT D$"BLOADPROGRAMM
A, A"ES
200  :
210  INPUT "STAMPO SU STAMPANTE? "; S$
222  INPUT "STAMPO LE VARIABILI REALI?
"; S1$
224  INPUT "STAMPO LE VARIABILI STRINGA
? "; S2$
226  INPUT "STAMPO LE VARIABILI INTERE?
"; S3$
228  INPUT "STAMPO GLI ARRAY REALI? "; S
4$
230  INPUT "STAMPO GLI ARRAY STRINGA? "
; S5$
232  INPUT "STAMPO GLI ARRAY INTERI? ";
S6$
233  IF SS4$ = "S" OR S6$ = "S" THEN  I
NPUT "STAMPO GLI ELEMENTI DI ARRAY NUME
RICI   CHE VALGONO ZERO? "; S8$
234  IF S2$ = "S" OR S5$ = "S" THEN  IN
PUT "STAMPO UN MESSAGGIO PER LE STRINGH
E   CONTENUTE NEL PROGRAMMA? "; S7$
235  IF S$ = "S" OR S$ = "" THEN  PRINT
 CHR$ (4)"PR#1":  PRINT  CHR$ (9)"80N
236  :
237  :
238  REM STAMPA DELLE VARIABILI SEMPLIC
I
239  :
240 I = SV
245  IF S1$ <  > "S" AND S2$ <  > "S" A
ND S3$ <  > "S" THEN I = SA:  GOTO 480
250  PRINT :  PRINT "   VARIABILI:":  PRIN
T
260  GOSUB 5000
360  IF  NOT FL THEN FL = 1
380  ON FL GOSUB 2000, 3000, 4000
400  IF SA > I THEN 260
430  :
440  :
450  REM STAMPA DEGLI ARRAY
460  :
480  PRINT :  PRINT :  PRINT "   ARRAY:":
PRINT
500  GOSUB 5000:  PRINT :  PRINT B$; :  REM
 NOME DELL'ARRAY
520 M = I +  PEEK (I) - 2 +  PEEK (I +
1) * 256: I = I + 2
530  PRINT C$(FL); : N =  PEEK (I): I = I
+ 1
535  REM STAMPA LE DIMENSIONI
540  FOR J = 1 TO N: A%(J) =  PEEK (I) *
 256 +  PEEK (I + 1) - 1: I = I + 2: B%(J
) = 0:  NEXT
560  PRINT  CHR$ (91); :  FOR J = N TO 1
STEP  - 1:  PRINT A%(J); :  IF J <  > 1 TH
EN  PRINT ",";
580  NEXT :  PRINT "]"
600  IF  NOT FL THEN FL = 1
620  IF FL = 1 AND S4$ <  > "S" OR FL =
 3 AND S6$ <  > "S" OR FL = 2 AND S5$ <
 > "S" AND S7$ <  > "S" THEN 800
630  REM STAMPA IL NUMERO DELL'ELEMENTO

640 C$ = "   (":  FOR K = N TO 1 STEP  -
1: C$ = C$ +  STR$ (B%(K)):  IF K <  > 1
THEN C$ = C$ + ","
660  NEXT : C$ = C$ + ") = "
670  REM STAMPA IL VALORE DELL'ELEMENTO
680  ON FL GOSUB 12000, 13000, 14000
700 K = 1:  REM NUMERO DELLA DIMENSIONE
720 B%(K) = B%(K) + 1:  REM NUMERO DELL'
ELEMENTO
740  IF K > N THEN 800
760  IF B%(K) > A%(K) THEN B%(K) = 0: K
= K + 1:  GOTO 720:  REM PASSA ALLA DIMEN
SIONE SUCCESSIVA
780  GOTO 640
800 I = M
820  IF EN > I THEN 500
840  PRINT D$"PR#0"
860  PRINT D$"EXECREPORTER. E
1960  :
1970  :
1980  REM STAMPA UNA VARIABILE REALE
1990  :
2000  IF S1$ <  > "S" THEN I = I + 5:  R
ETURN
2020  GOSUB 20000:  PRINT B$" = "T:  RET
URN
2960  :
2970  :
2980  REM STAMPA UNA VARIABILE STRINGA
2990  :
3000  IF S2$ <  > "S" THEN I = I + 5:  R
ETURN
3010 L =  PEEK (I): I = I + 1
3020 ST =  PEEK (I) +  PEEK (1 + I) * 2
56
3025 C$ = B$ + "$ = "
3028  IF  NOT (L AND ST) THEN I = I + 4
```

```
: RETURN
3030  GOSUB 30000: I = I + 4:  RETURN
3960 :
3970 :
3980  REM STAMPA UNA VARIABILE INTERA
3990 :
4000  IF S3$ <  > "S" THEN I = I + 5: R
ETURN
4010  GOSUB 40000: PRINT B$"% = "T: I =
I + 5: RETURN
4960 :
4970 :
4980  REM TROVA IL NOME DELLA VARIABILE

4990 :
5000 FL = 0: A =  PEEK (I): I = I + 1: IF
 A > 127 THEN A = A - 128: FL = 1
5020 B$ =  CHR$ (A)
5040 A =  PEEK (I): I = I + 1: IF A > 12
7 THEN A = A - 128: FL = FL + 2
5060 B$ = B$ +  CHR$ (A): RETURN
11960 :
11970 :
11980  REM STAMPA UN ELEMENTO DI ARRAY
REALE
11990 :
12000  IF S4$ <  > "S" THEN I = I + 5:
RETURN
12020  GOSUB 20000: IF S8$ < > "S" AND
 NOT T THEN  RETURN
12040  PRINT C$T: RETURN
12960 :
12970 :
12980  REM STAMPA UN ELEMENTO DI ARRAY
STRINGA
12990 :
13000  IF S5$ <  > "S" THEN I = I + 3:
RETURN
13010 L =  PEEK (I): I = I + 1
13020 ST =  PEEK (I) +  PEEK (I + 1) *
256
13040  IF  NOT (L AND ST) THEN I = I +
2: RETURN
13060  GOSUB 30000: I = I + 2: RETURN
13960 :
13970 :
13980  REM STAMPA UN ELEMENTO DI ARRAY
INTERO
13990 :
14000  IF S6$ <  > "S" THEN I = I + 2:
RETURN
14020  GOSUB 40000: IF S8$ <  > "S" AND
  NOT T THEN  RETURN
14040  PRINT C$T: RETURN
19960 :
19970 :
19980  REM CALCOLA IL VALORE DI UN NUME
RO REALE
19990 :
20000 A =  PEEK (I): I = I + 2
20020 T = 2 ^ (A - 129)
20040 T = T +  PEEK (I) * 2 ^ (A- 144)
: I = I + 1
20060 T = T +  PEEK (I) * 2 ^ (A - 152)
: I = I + 1
20080 T = T +  PEEK (I) * 2 ^ (A- 160)
: I = I + 1
20100  IF  PEEK (I - 4) < 128 THEN T =
T +  PEEK (I - 4) * 2 ^ (A - 136)
20120  IF  PEEK (I - 4) >127 THEN T =
 - T - ( PEEK (I - 4) - 128) * 2 ^ (A -
 136)
20140  RETURN
29960 :
29970 :
29980  REM CALCOLA IL VALORE DI UNA STR
INGA
29990 :
30000 A$ = "": FG = 0: ST = ST + D2
30020 B$ = ""
30200  IF ST < 1024 + D2 THEN B$ = "POI
NTER PRIMA DEL PROGRAMMA": GOTO 30500
30205  IF ST < SV AND S7$ = "S" THEN B$
 = "STRINGA CONTENUTA NEL PROGRAMMA
30210  IF ST < SV THEN ST = ST - D2 + D
3: GOTO 30260
30220  IF ST < SS THEN B$ = "POINTER SO
TTO LO STRINGPOOL": GOTO 30500
30230  IF ES < ST THEN B$ = "POINTER SO
PRA HIMEM": GOTO 30500
30260  FOR S = ST TO ST + L - 1
30280 A =  PEEK (S): IF A = 12 THEN A =
7
30300 A$ = A$ +  CHR$ (A)
30320  IF A > 127 THEN FG = FG + 1
30330  NEXT
30340  PRINT C$ CHR$ (34)A$ CHR$ (34);:
 HTAB 40: INVERSE : PRINT B$: NORMAL
30350  IF FG = 0 THEN 30400
30355  PRINT "CODICI ASCII: ";
30360  FOR S = ST TO ST + L - 1: A =  PE
EK (S): PRINT A" ";
30380  NEXT : PRINT
30400  RETURN
30500  IF S7$ <  > "S" THEN  RETURN
30510  PRINT C$B$"; INDIRIZZO DEL POINT
ER ="I - D1
30520  RETURN
39960 :
39970 :
39980  REM CALCOLA IL VALORE DI UN NUME
RO INTERO
39990 :
40000 T =  PEEK (I) * 256 +  PEEK (I +
1)
40020  IF T > 32767 THEN T = T - 65536
40040  RETURN
```

Figura 2

```
 ┌───────────────────┐
 :                   :  PUNTATORI DEL
 | 105   -   117     |    PROGRAMMA
 :                   :    REPORTER
 ├───────────────────┤
 | (2049 ($801)      |    PROGRAMMA
 :                   :    REPORTER
 |     LOMEM         |
 ├───────────────────┤
 |     LOMEM+1       |  VARIABILI DEL
 :                   :    PROGRAMMA
 |   9987 ($2703)    |    REPORTER
 ├───────────────────┤
 |   9988 ($2704)    |  PUNTATORI DEL
 :                   :    PROGRAMMA
 |   9999 ($270F)    |    IN SVILUPPO
 ├───────────────────┤
 |  10000 ($2710)    |  VARIABILI DEL
 :                   :    PROGRAMMA
 |       EN          |    IN SVILUPPO
 ├───────────────────┤
 |       SS          |  STRINGPOOL DEL
 :                   :    PROGRAMMA
 |       ES          |    IN SVILUPPO
 ├───────────────────┤
 |      ES+1         |    PROGRAMMA
 :                   :    IN SVILUPPO
 |                   |
 ├───────────────────┤
 |                   |     MEMORIA
 :                   :     LIBERA
 |  36864 ($9000)    |
 └───────────────────┘
```

Figura 3

```
10 A$ = "PROVA"
20 B$ = A$
30 A$ = A$ + ""
```

Figura 4

```
BLOADPROGRAMMA
?CHR$(4)"BLOADPUNTA,A105
?CHR$(4)"BLOADNUMERI,A"PEEK(106)*256+PEEK(105)"
?CHR$(4)"BLOADSTRINGHE,A"PEEK(112)*256+PEEK(111)"
?CHR$(4)"BLOADPUNTA,A105
```

Figura 5

```
100 D$ =  CHR$ (4)
120  PRINT D$"OPENREPORTER
140  PRINT D$"WRITEREPORTER
160  PRINT "?CHR$(4)" CHR$ (34)"BSAVEPU
NTA, A105, L12"
180  PRINT "?CHR$(4)" CHR$ (34)"BSAVENU
MERI, A" CHR$ (34)"PEEK(106)*256+PEEK(10
5)" CHR$ (34)", L" CHR$ (34)"(PEEK(110)*
256+PEEK(109)-PEEK(106)*256-PEEK(105))"

200  PRINT "?CHR$(4)" CHR$ (34)"BSAVEST
RINGHE, A" CHR$ (34)"PEEK(112)*256+PEEK(
111)" CHR$ (34)", L" CHR$ (34)"(PEEK(116
)*256+PEEK(115)-PEEK(112)*256-PEEK(111)
)"
220  PRINT "?CHR$(4)" CHR$ (34)"BSAVEPR
OGRAMMA ,A1024, L" CHR$ (34)" PEEK (106)
 * 256 +  PEEK (105) - 1024"
240  PRINT "RUNREPORTER. P"
260  PRINT D$"CLOSEREPORTER
```

▲ *Figura 6*

```
100 D$ =  CHR$ (4)
120  PRINT D$"OPENREPORTER. E
140  PRINT D$"WRITEREPORTER. E
150  PRINT "BLOADPROGRAMMA
160  PRINT "?CHR$(4)" CHR$ (34)"BLOADPU
NTA, A105
180  PRINT "?CHR$(4)" CHR$ (34)"BLOADNU
MERI, A" CHR$ (34)"PEEK(106)*256+PEEK(10
5)" CHR$ (34)
200  PRINT "?CHR$(4)" CHR$ (34)"BLOADST
RINGHE, A" CHR$ (34)"PEEK(112)*256+PEEK(
111)" CHR$ (34)"
220  PRINT "?CHR$(4)" CHR$ (34)"BLOADPU
NTA, A105
260  PRINT D$"CLOSEREPORTER. E
```

◀ *Figura 7*

Calendario cosmico

CAI: apprendiamo la storia e le scienze naturali.

di **G. U. Barzaghi**

Se paragonata all'età della Terra, che è infinitamente vecchia, l'umanità appare relativamente giovane.
Gli avvenimenti principali della nostra vita si possono misurare in anni, o, addirittura, in mesi; la nostra esistenza in decenni; la storia della propria famiglia in secoli; e l'intera storia documentabile in migliaia di anni.
Prima della comparsa dell'uomo sulla terra, però, è trascorso un periodo di tempo abissale, la cui durata è ammantata dalle nebbie che circondano il più remoto passato del nostro pianeta e sul quale disponiamo di una quantità di informazioni ridottissima, sia per scarsità di documentazione, sia per l'impossibilità di comprenderne appieno la vastità.
Nonostante ciò, disponiamo, a tutt'oggi, della possibilità di datazioni relativamente precise anche per eventi di epoche antichissime.
L'esame degli strati geologici ed il metodo di datazione basato sul contenuto di radioattività dei reperti (l'esame al Carbonio 14) forniscono informazioni utili all'archeologia, alla paleontologia e alla geologia; con i dati che gli astrofisici si sono procurati è possibile stabilire le età dei pianeti, delle stelle, della nostra galassia (la Via Lattea) ed inoltre calcolare, con buona approssimazione, il tempo trascorso dall'avvenimento più remoto di cui si abbia traccia: il "famigerato" "Big Bang" (la enorme - ed in questo caso è difficile stabilire un ordine di grandezza per questa enormità - esplosione di materia gassosa cosmica ad altissima densità che, secondo una teoria ormai generalmente accettata, avrebbe dato origine all'universo circa 15 miliardi di anni fa), di cui conserviamo un singolare reperto sotto forma di "radiazione fossile", captabile con antenne di particolare sensibilità, e che altro non sarebbe che l'ultimo eco del rumore provocato dall'esplosione stessa.
Il modo più chiaro ed efficace che si conosca, secondo quanto afferma Carl Sagan, professore di astronomia e scienza dello spazio alla Cornell University, nel suo libro "The dragons of Eden", per fornire una visione sinottica di questa cronologia cosmica, consiste nell'immaginare che i 15 miliardi di anni di vita dell'universo siano compresi nell'arco di un solo anno solare.
Tramite questa ideale abbreviazione, ogni miliardo di anni di vita della Terra può venire completamente rappresentato in un intervallo coincidente con, all'incirca, 24 giorni di quello che chiameremo, d'ora in poi, "anno cosmico".
Il calendario cosmico è suddiviso in tre sezioni: le date principali per il periodo Gennaio-Novembre (che chiameremo, convenzionalmente, "anno cosmico"); i fatti del mese di Dicembre seguiti quasi giorno per giorno; ed un più minuzioso esame delle ultime ore dell'ultimo giorno dell'anno.
Lavorando a calendari e tavole sinottiche di questo tipo è inevitabile sentirsi piccoli e privi di importanza.
E' ad esempio interessante notare che in un simile anno cosmico, la materia interstellare che darà origine alla Terra si condensa solo alla metà di Settembre; i dinosauri compaiono alla vigilia di Natale; la specie umana appare sulla scena alle 22.30 dell'ultimo giorno dell'anno; l'intera storia documentata è compresa negli ultimi dieci secondi del 31 Dicembre e il tempo trascorso dalla fine del Medioevo ad oggi dura poco più di un secondo.
Eppure, nonostante il fatto che la parte fino ad oggi occupata dalla specie umana del tempo cosmico sia infinitesima, è evidente che tutto ciò che accadrà sul nostro pianeta e nello spazio a noi accessibile a partire dal secondo anno cosmico dipenderà in larga misura dall'uso che l'umanità saprà fare della scienza e di quella razionalità che dovrebbe essere una delle sue più spiccate caratteristiche.
Proprio per sottolineare i pericoli e le ambiguità insite nel progresso e nello sviluppo tecnologico proprio della nostra epoca, ho proposto, per identificare il periodo che stiamo vivendo, due alternative: da un lato gli aspetti positivi della nostra epoca - lo sviluppo della scienza e della tecnologia; l'affermarsi *di una cultura* planetaria, l'esplorazione dello spazio -; dall'altro l'acquisizione, da parte della

CBM

Calendario cosmico

specie umana dei mezzi di auto-distruzione, il pericolo dell'olocausto nucleare. Sta al lettore - in base al suo pessimismo o alla sua fiducia in un futuro migliore - decidere quale dei due aspetti caratterizza meglio questi nostri anni di piombo.

Il programma

Il programma in questione rappresenta un semplice esempio di gioco di simulazione per l'apprendimento tramite calcolatore. Il gioco intende correggere quanto c'è di negativo nell'insegnamento tradizionale della storia e delle scienze naturali, creando una situazione favorevole all'apprendimento ed alla verifica di quanto spesso l'utente sa, ma senza collegare con precisione gli avvenimenti e le date e favorendo la collocazione delle nozioni che ne emergono in uno schema di riferimento che ne amplii il significato.

L'utente può, innanzi tutto, stabilire la percentuale di errore massima ammissibile per le tre sezioni in cui il gioco è suddiviso. In questo modo è possibile, anche per chi non avesse una precisa idea degli argomenti trattati, utilizzare il programma come metodo di apprendimento degli argomenti stessi, iniziando a giocare con percentuali massime di errore alte (66%, vale a dire una sola risposta giusta su tre), diminuendole via via fino alla minima (nessun errore ammissibile). Il programma provvede infatti a far ripetere all'utente la sezione in questione finchè la sua percentuale di errore per quella sezione non è inferiore alla massima ammissibile da lui scelta.

Sul video appaiono, in ordine casuale e variabile, gli avvenimenti raggruppati nelle tre sezioni di cui si è detto nella prima parte dell'articolo, nella parte inferiore della pagina video comparе una data della stessa sezione, associabile all'avvenimento suddetto. Il giocatore, se ritiene che l'avvenimento e la data coincidano, può indicarlo premendo il tasto *C*, oppure, se ritiene che l'associazione sia errata, richiede una nuova data, premendo il tasto *N*.

Il calcolatore stesso provvederà a sottolineare i vostri errori, a scopo educativo, evidenziando in negativo la data erroneamente associata all'avvenimento; al termine della sezione, come detto, se la percentuale di errori sarà superiore a quella massima da voi indicata, vi verrà riproposta la stessa sezione, in modo da correggere i vostri errori.

Ad ognuna delle tre parti è associato un orologio, che indica il tempo parziale impiegato per ognuna; mentre un orologio centrale dà il tempo totale di gioco.

Essendo lo scopo del gioco essenzialmente educativo non si è prevista una possibilità di utilizzo competitivo, tra più giocatori, del programma stesso. Pur essendo sempre stato un acceso fautore dello spirito di emulazione nell'apprendimento, devo ammettere che esso è, almeno a volte, contro producente.

Gli aspetti tecnici

Il programma ha un'occupazione di memoria limitata e non presenta eccessive difficoltà. Personalmente vorrei sottolineare solo un paio di aspetti che rendono interessante il programma da un punto di vista puramente informatico.

Innanzi tutto le frasi *DATA* da riga *20* a riga *63*: come potete notare, le stringhe contenute nelle frasi *DATA* suddette contengono un insolito numero di spazi bianchi; ciò è dovuto al fatto che l'uscita su video delle stringhe stesse (che rappresentano gli avvenimenti del nostro calendario cosmico) non viene giustificata da programma. E' quindi necessario disporre opportunamente le stringhe in modo da ottenere una uscita sul video congruente e corretta anche da un punto di vista grammaticale.

Pertanto il lettore deve porre la massima attenzione nel ricopiare le righe suddette, contando accuratamente gli spazi bianchi tra le stringhe alfanumeriche e non modificandone minimamente la disposizione, per quanto originale o stravagante possa apparire.

Dato che si è fatto uso del minuscolo, a causa dell'abbondanza di stringhe di testo nel programma, nel listato compaiono, invece di caratteri alfanumerici, dei simboli grafici, ciò è dovuto alla presenza, in corrispondenza di quei simboli delle lettere maiuscole corrispondenti ai tasti dei caratteri in questione.

Un altro aspetto interessante del programma è rappresentato dalla indicizzazione della funzione generatrice di numeri pseudo-casuali *RND*, nella subroutine di mescolamento *500-530*. L'indice *D* inseritovi genera serie di numeri pseudocasuali uguali ad uguale valore dell'indice *D* stesso. Non si è fatto uso, in questo caso, dell'indice *TI* (tempo trascorso dall'accensione del sistema) o *-TI*, poichè, essendo le due istruzioni *F1=INT(RND(D)★D(D)★D(D)+1)* e *F2=INT(RND(D)★D(D)+1)* immediatamente successive, il tempo *TI* (in sessantesimi di secondo!) non cambiava apprezzabilmente da un'istruzione all'altra, ed il risultato della funzione *RND* era quindi identico: essendo le istruzioni in questione responsabili della scelta degli indici dei due elementi della tabella che vengono scambiati tra loro, per mescolare gli elementi della tabella stessa, il risultato era, in realtà, di ritrovarsi con la tabella nell'ordine esatto in cui era stata letta dalle frasi *DATA*.

L'ultimo aspetto particolare riguarda l'editing di video. Come i lettori più fedeli avranno già avuto modo di rilevare, manca in questo programma il "famigerato" vettore *W$* che, in quasi tutti gli altri miei programmi, era responsabile del corretto posizionamento *dei caratteri* sul video.

Questa assenza non è dovuta ad un ripensamento circa la sua efficacia od utilità, ma all'inserimento, fatto a bella posta, di una funzione avente analoghi effetti, anche se più gravosa come applicazione ed occupazione di memoria, che era mia intenzione mostrarvi.

La funzione in questione è rappresentata dalla subroutine *1000*: in essa la variabile *R1* identifica la riga su cui si vuole rappresentare un particolare carattere, la variabile *C* la colonna a partire dalla quale va inserita su video la stringa che si vuole posizionare, mentre l'istruzione *SYS57949* attiva l'effetto voluto. Questa serie di istruzioni deve essere seguita da una *PRINT* e dalla variabile numerica od alfanumerica desiderata.

La forma da me utilizzata nel corso del programma, permette di richiamare da più punti la funzione suddetta, passando di volta in volta come variabili gli indici di riga e di colonna e facendo seguire ad un accesso alla subroutine l'istruzione di *PRINT* corrispondente. ■

REMarks

0	*Apertura del minuscolo e inizializzazione di alcune variabili.*
5	*Inizializzazione di altre variabili e dimensionamento delle matrici: A$(17,2,3), contenente gli avvenimenti delle tre sezioni del calendario e le date corrispondenti (la matrice è surdimensionata).* *A(17,3,3), contenente delle variabili numeriche corrispondenti agli elementi della matrice suddetta ed indicanti la correttezza o meno della risposta dell'utente ed il fatto che una data sia stata già utilizzata.* *S$(3,3), contenente i tempi parziali per le tre sezioni di gara, suddivisi in ore, minuti primi e minuti secondi.*
10	*Inizializzazione della matrice S$ ed accesso alla subroutine responsabile della scelta della percentuale massima di errore ammissibile.*
20-63	*Frasi DATA contenenti gli avvenimenti del calendario cosmico e le date ad essi associate.*
100	*Lettura delle frasi DATA ed inizializzazione con valori opportuni della matrice A. Notare la chiu-*

	sura simultanea dei tre cicli FOR ... NEXT, ottenuta elencando ordinatamente gli indici dei vari cicli, a partire dal più interno al più esterno, separandoli con virgole.
103-111	*Display della base di gioco. In particolare:*
100	*Cornice degli orologi parziali.*
106	*Cornice dell'orologio totale.*
109	*Inizializzazione dell'orologio interno del calcolatore.*
140	*Accesso alla subroutine di mescolamento, inizializzazione della variabile contenente il punteggio parziale per la funzione in corso e clearing della sezione di video occupata dalle stringhe alfanumeriche degli avvenimenti (GOSUB2000) e aggiornamento degli orologi (GOSUB3000).*
150	*Display dell'E-esimo avvenimento della D-esima sezione. L'inizializzazione della variabile N a 1 consente di iniziare a scandire la matrice delle date a discrezione dell'utente.*
170	*Se il campo di indice 3 associato alla N-esima data della D-esima sezione contiene un valore negativo, indicante che è già stata accoppiata (a torto o a ragione) con un avvenimento, si salta ad una opportuna sezione.*
175	*Clearing della sezione di video riservata alle date e display di una nuova data.*
180	*Acquisizione di caratteri da tastiera (tramite una GET, è perciò superfluo premere il tasto RETURN). In assenza di tasti premuti si ricicla.*
190	*Se il tasto premuto è C si salta ad una sezione opportuna.*
193	*Se il tasto premuto non è N (vale a dire è un tasto premuto per errore) si ricicla per un nuovo carattere.*
195	*Aggiornamento della variabile di scansione della matrice delle date, aggiornamento degli orologi; se la variabile N assume un valore superiore al limite massimo per la matrice in questione, si scandisce di nuovo l'array a partire dall'inizio.*
197	*Aggiornamento degli orologi e trasferimento del flusso del programma ad un punto opportuno per una nuova data.*
200	*Se è stata scelta una coppia avvenimento-data e la scelta è stata esatta si aggiorna il punteggio.*
203	*Accesso alla subroutine che segnala gli errori di scelta.*
205	*Eliminazione della scansione delle date già accoppiate e chiusura del ciclo. Al termine del ciclo si controlla se la percentuale di errori è superiore a quella massima prescelta, se la risposta è positiva si ripete l'ultima sezione percorsa.*
210	*Chiusura del ciclo più esterno (quello sulle tre sezioni).*
220-230	*Fine prova.*
500-520	*Subroutine di mescolamento. Si approfitta della subroutine in questione anche per ri-inizializzare opportunamente le variabili indicanti se una data è già stata utilizzata o no. Ciò poichè la matrice corrispondente viene rimescolata prima di ogni scansione della stessa sezione ANCHE in caso di ripetizione della sezione stessa. Il mescolamento è effettuato scegliendo, a caso - tramite la funzione RND -, due elementi della matrice di cui viene scambiata la posizione.*
1000	*Subroutine di posizionamento delle stringhe sul video. La sezione in questione è già stata diffusamente descritta nell'articolo.*
1500	*Questo sottoprogramma è responsabile della inversione nella notazione della data, in caso di errore: il ciclo FOR ... NEXT scandisce la parte di video occupata dalla data, la frase PEEK consente di ottenere il numero di POKE del carattere di indice video P, aggiungendo a questo numero di POKE il valore numerico 128, si ottiene la notazione inversa del carattere in questione. Il carattere, in notazione inversa, viene impresso - anzi sovraimpresso - nella posizione di indice P, tramite la frase POKE P,R.*
2000	*Subroutine di clearing della sezione di video riservata agli avvenimenti del calendario cosmico.*
2500	*Subroutine di clearing della sezione di video dedicata alle date del calendario cosmico.*
3000-3080	*Subroutine di aggiornamento degli orologi. In particolare:*
3000	*Dall'orologio interno del calcolatore vengono opportunamente ricavati ed attribuiti alle variabili Z$ le ore, i minuti primi ed i minuti secondi trascorsi dall'inizio del test.*
3010	*Nel caso in cui il giocatore sta affrontando la prima sezione, l'orologio parziale coincide con quello totale: rendendo perciò superflua ogni successiva elaborazione.*
3020	*Si ricava, per le sezioni successive alla prima, la differenza tra il tempo totale ed i tempi parziali delle sezioni precedenti, eventualmente cumulati.*
3030	*Se la differenza dell'istruzione di cui sopra dà risultato negativo, vengono effettuate le opportune correzioni, sommando 60 al risultato e diminuendo la grandezza immediatamente precedente dell'unità presa a prestito, come da operazioni con grandezze sessagesimali.*
3050-3080	*Gli orologi vengono opportunamente aggiornati. In 3050-3060 l'orologio della sezione di gioco in corso. In 3070-3075 quello generale.*
3500-3510	*Sottoprogramma di display del segnale di errato ordinamento. Scatta quando la sezione in corso ha visto una percentuale di errori superiore a quella massima ammessa.*
4000-4057	*Subroutine responsabile della scelta della percentuale massima di errore ammissibile.*

***Figura 1** - Le principali parti in cui è suddiviso il programma.*

```
CALENDARIO COSMICO

 0 POKE59468,14:A=9:B=17:C=13:D(1)=A:D(
2)=B:D(3)=C:R=11:C=4:T$(1)=" ┌──────────┐"
 5 T$(2)="|00:00:00|":T$(3)=" └──────────┘
":DIMA$(17,2,3),A(17,3,3),S$(3,3)
 10 FORD=1TO3:FORE=1TO3:S$(D,E)="00":NE
XTE,D:PRINT"□":GOSUB4000
 20 DATA"1 |ENNAIO","'|IG |ANG' (GRANDE
 ESPLOSIONE)","1 \AGGIO"
 21 DATA"┌RIGINE ×IA └ATTEA","9 ♦ETTEMB
RE","┌RIGINE DEL ♦ISTEMA ♦OLARE"
 22 DATA"14 ♦ETTEMBRE","—ORMAZIONE DELL
A |ERRA","25 ♦ETTEMBRE"
 23 DATA"┌RIGINE DELLA VITA SULLA |ERRA
","9 ┌TTOBRE"
 24 DATA"—ATAZIONE DEI FOSSILI PIU' ANT
ICHI","1 ╱OVEMBRE"
 25 DATA"—OMPARSA DI MICRORGANISMI SESS
UATI","12 ╱OVEMBRE"
 26 DATA"—ATAZIONE DELLE PIU' ANTICHE P
IANTE      FOTOSINTETICHE FOSSILI"
 27 DATA"15 ╱OVEMBRE","┐RIME CELLULE FO
RNITE DI NUCLEO","└UNEDI' 1"
 28 DATA"╱N' ATMOSFERA RICCA DI OSSIGEN
O SI       FORMA ATTORNO ALLA |ERRA"
 29 DATA"×ENERDI' 5","—IFFUSO VULCANISM
O E FORMAZIONE DEI      'CANALI' SU \ART
E"
 30 DATA"\ARTEDI' 16","┐RIMI VERMI","\E
RCOLEDI' 17"
 31 DATA"—INE ┐RECAMBRIANO, INIZIO ┐RA
┐ALEO      ZOICA, COMPARSA INVERTEBRATI"
 32 DATA"|IOVEDI' 18","┐RIMO PLANCTON O
CEANICO","×ENERDI' 19"
 33 DATA"┐ERIODO ┌RDOVICIANO; PRIMI PES
CI,        PRIMI VERTEBRATI","♦ABATO 20"
 34 DATA"┐ERIODO ♦ILURIANO; PRIME PIANT
E VA-      SCOLARI; VEGETAZIONE DIFFUSA"
 35 DATA"—OMENICA 21"
 36 DATA"┐ERIODO —EVONIANO; PRIMI INSET
TI;        ANIMALI INVADONO TERRA EMERSA
"
```

Calendario cosmico

```
 37 DATA"LUNEDI' 22","PRIMI ANFIBI; PRI
MI INSETTI ALATI","MARTEDI' 23"
 38 DATA"PERIODO CARBONIFERO; PRIMI ALB
ERI;      PRIMI RETTILI","MERCOLEDI' 24
"
 39 DATA"COMINCIA IL PERIODO PERMIANO;
PRIMI       DINOSAURI","VENERDI' 26"
 40 DATA"PERIODO TRIASSICO; PRIMI MAMMI
FERI","SABATO 27"
 41 DATA"PERIODO GIURASSICO; PRIMI UCCE
LLI","DOMENICA 28"
 42 DATA"PERIODO CRETACEO; PRIMI FIORI;
 I DI       NOSAURI SI ESTINGUONO"
 43 DATA"LUNEDI' 29"
 44 DATA"FINE ERA MESOZOICA; INIZIO ERA
 CENO       ZOICA;TERZIARIO; PRIMI PRIMAT
I"
 45 DATA"MARTEDI' 30"
 46 DATA"EVOLUZIONE LOBI FRONTALI NEI P
RIMA-       TI; OMINIDI; MAMMIFERI GIGANT
I"
 47 DATA"MERCOLEDI' 31","FINE DEL PLIOC
ENE","ORE 22.30'"
 48 DATA"PRIMI ESSERI UMANI","ORE 23.00
'"
 49 DATA"USO GENERALIZZATO DEGLI UTENSI
LI DI       PIETRA","ORE 23.46'"
 50 DATA"CONTROLLO DEL FUOCO DA PARTE D
EL-         L'UOMO DI PECHINO"
 51 DATA"ORE 23.56'","INIZIO DELLA PIU'
 RECENTE EPOCA GLA      CIALE"
 52 DATA"ORE 23.59'"
 53 DATA"PITTURE RUPESTRI IN EUROPA","O
RE 23.59'20''","NASCE L'AGRICOLTURA"
 54 DATA"ORE 23.59'51''","INVENZIONE DE
LL'ALFABETO","ORE 23.59'53''"
 55 DATA"ETA' DEL BRONZO; CULTURA MICEN
EA;         GUERRA DI TROIA","ORE 23.59'5
5"
 56 DATA"ATENE NELL'ETA' DI PERICLE; NA
SCI-        TA DI BUDDA","ORE 23.59'56''
"
 57 DATA"GEOMETRIA EUCLIDEA; ARCHIMEDE;
 IM-        PERO ROMANO; NASCITA DI CRIST
O"
 58 DATA"ORE 23.59'58''"
 59 DATA"CIVILTA' MAYA; IMPERO BIZANTIN
O;          INVASIONI MONGOLE; CROCIATE"
 60 DATA"ORE 23.59'59''"
 61 DATA"RINASCIMENTO; METODO SCIENTIFI
CO          SPERIMENTALE"
 62 DATA"OGGI: PRIMO SECONDO DI CAPODAN
NO DEL      NUOVO ANNO COSMICO"
 63 DATA"ACQUISIZIONE DEI MEZZI DI AUTO
 DI-        STRUZIONE DA PARTE DELL'UOMO"
 100 FORD=1TO3:FORE=1TOD(D):FORF=1TO2:R
EADA$(E,F,D):A(E,F,D)=E:NEXTF,E,D
 103 FORH=1TO3:FORG=1TO3:PRINTTAB(3+(G-
1)*12)T$(H);:NEXT:PRINT
 105 NEXT:PRINTTAB(6)"ANNO"TAB(16)"DICE
MBRE"TAB(31)"31"
```

```
106 PRINTTAB(4)"-OSMICO"TAB(28)"-ICEMB
RE":FORH=1TO3:PRINTTAB(15)T$(H):NEXT
108 PRINTTAB(17)"|OTALE":R1=R:C=3:GOSU
B1000:PRINT"♠VVENIMENTO:":R1=17:GOSUB10
00
109 PRINT"-ATA CORRISPONDENTE NEL CALE
NDARIO       COSMICO:":TI$="000000"
110 FORD=1TO3:R1=22:C=3:GOSUB1000:PRIN
T"'C' SE LA DATA COINCIDE"
111 PRINTTAB(3)"'N' PER UN'ALTRA DATA"
140 GOSUB500:P(D)=0:FORE=1TOD(D):GOSUB
2000:GOSUB3000:R1=13:C=3:GOSUB1000
150 PRINTA$(E,2,D):N=1
170 IFA(N,3,D)<0THEN195
175 GOSUB2500:GOSUB3000:R1=20:C=3:GOSU
B1000:PRINTA$(N,1,D)
180 GOSUB3000:GETR$:IFR$=""THEN180
190 IFR$="C"THEN200
193 IFR$<>"N"THEN180
195 N=N+1:IFN>D(D)THENN=1
197 GOSUB3000:GOTO170
200 GOSUB3000:IFA(E,2,D)=A(N,1,D)THENP
(D)=P(D)+1:GOTO205
203 GOSUB1500:GOSUB3000
205 A(N,3,D)=-1:GOSUB3000:NEXT:IFP(D)<
D(D)*PCTHENGOSUB3500:GOTO140
210 GOSUB3000:NEXT
220 FORP=33168TO34768:POKEP,32:NEXT:R1
=15:C=5:GOSUB1000
230 PRINT"┌RDINAMENTO CORRETTO":FORP=1
TO50000:NEXT:END
500 FORE=1TOD(D):A(E,3,D)=1:NEXT:FORE=
1TOD(D)*3:F1=INT(RND(D)*D(D)+1)
510 F2=INT(RND(D)*D(D)+1):A$=A$(F1,2,D
):AA=A(F1,2,D):A$(F1,2,D)=A$(F2,2,D)
520 A(F1,2,D)=A(F2,2,D):A$(F2,2,D)=A$:
A(F2,2,D)=AA:NEXT:RETURN
1000 POKE216,R1:POKE198,C:SYS57949:RET
URN
1500 FORP=33571TO33605:R=PEEK(P):R=R+1
28:POKEP,R:NEXT:RETURN
2000 FORP=33248TO33367:POKEP,32:NEXT:R
ETURN
2500 FORP=33528TO33648:POKEP,32:NEXT:R
ETURN
3000 Z$(1)=LEFT$(TI$,2):Z$(2)=MID$(TI$
,3,2):Z$(3)=RIGHT$(TI$,2)
3010 IFD=1THENFORG=1TO3:S$(D,G)=Z$(G):
NEXT:K=0:GOTO3050
3020 FORG=3TO1STEP-1:S=VAL(Z$(G))-VAL(
S$(D-1,G))
3030 IFS<0THENS=S+60:Z$(G-1)=RIGHT$("0
0"+STR$(VAL(Z$(G-1))-1),2)
3040 S$(D,G)=STR$(S):NEXT:K=1
3050 R1=2:C=4:GOSUB1000:FORH=1TO3:T=4+
(D-1)*12
3055 S$(D,H)=RIGHT$(S$(D,H),LEN(S$(D,H
))-K)
3060 X$=RIGHT$("00"+S$(D,H),2):PRINTTA
B(T+(H-1)*3)X$;:NEXT
3070 R1=7:C=0:GOSUB1000:FORG=1TO3:Z$(G
)=RIGHT$(Z$(G),LEN(Z$(G)))
3075 X$=RIGHT$("00"+Z$(G),2):PRINTTAB(
16+(G-1)*3)X$;:NEXT
3080 RETURN
3500 GOSUB2000:GOSUB2500:R1=14:C=12:GO
SUB1000:PRINT"▣RDINAMENTO ERRATO■"
3510 FORP=1TO1000:NEXT:GOSUB2500:RETUR
N
4000 R1=5:C=3:GOSUB1000:PRINT"♥CEGLI L
A PERCENTUALE DI ERRORE AM-"
4010 PRINTTAB(3)"MISSIBILE SU OGNI SEZ
IONE:"
4015 PRINT:PRINT" 1 - NESSUN ERRORE AM
MESSO"
4020 PRINT:PRINT" 2 - 10%":PRINT:PRINT
" 3 - 20%":PRINT:PRINT" 4 - 25%"
4030 PRINT:PRINT" 5 - 40%":PRINT:PRINT
" 6 - 50%":PRINT:PRINT" 7 - 66%"
4040 PRINT:PRINT"▣_ISPOSTA:";
4045 INPUTW:IFW<1ORW>7THENPRINT"]]";:G
OTO4040
4046 PRINT"▣"
4050 ONWGOTO4051,4052,4053,4054,4055,4
056,4057
4051 PC=1:RETURN
4052 PC=0.9:RETURN
4053 PC=0.8:RETURN
4054 PC=0.75:RETURN
4055 PC=0.6:RETURN
4056 PC=0.5:RETURN
4057 PC=0.33:RETURN

Lista simboli grafici

  10 : 1  SHIFT HOME =CHR$(147)

3500 : 1  REVERSE =CHR$(18)
       1  SHIFT REVERSE =CHR$(146)

4040 : 1  REVERSE =CHR$(18)

4045 : 2  SHIFT CRSR↑ =CHR$(145)

4046 : 1  SHIFT HOME =CHR$(147)
```

Figura 2 - Il listato per il CBM.

Programmi binari per gli HP serie 80

Ordinamenti rapidissimi di stringhe alfabetiche, e altri

di **V. Anselmo**

Programmare bene non è facile. Intendiamoci: preparare un programmino in BASIC senza pretese di compattezza e velocità (basta che giri e mi dia i risultati) è alla portata di tutti. Scrivere programmi potenti, veloci e brevi in un linguaggio tipo Assembler richiede invece esperienza e mesi di lavoro. Per questo motivo, quando dalla macchina si vogliono ottenere risultati professionali e se ne vogliono sfruttare appieno le possibilità, sarà bene servirsi di programmi ampiamente sperimentati. Con l'uso di programmi binari che possono essere caricati in macchina assieme ad un programma principale in BASIC, si potranno anche arricchire e rendere più veloci e compatti i programmi che intendiamo preparare da soli.

I Personal Computer HP della Serie 80 (HP83, HP85, HP86 e HP87) hanno a disposizione da una parte tutta una serie di package preparati appositamente dalla Hewlett Packard e da Software House, dall'altra i programmi della UPLE (Users' Program Library Europe), una biblioteca di programmi sviluppati da utenti europei o, ancora dalla Hewlett Packard. Tra questi ultimi è ora disponibile una serie di programmi binari (scritti cioè in linguaggio macchina), che possono essere caricati nel calcolatore contemporaneamente ad un programma in BASIC. Come si sa, nell'HP-83/HP85 si può caricare un solo programma binario alla volta, oltre a quello in BASIC, mentre nell'HP-86/HP87 di tali programmi se ne possono caricare finora cinque. Tenuta presente la grande capacità di ampliamento della memoria centrale di queste ultime macchine (640 Kbyte per l'HP87 e 576 Kbyte per l'HP86),tale possibilità diventa estremamente interessante. Vediamo brevemente il contenuto di alcuni dei più interessanti programmi della UPLE. A molti sarà successo di dover operare la fusione (merge) di due programmi in uno. La cosa si può effettuare ribattendo le istruzioni (opportunamente rinumerate) di uno dei due programmi, oppure caricando il programma A, listarne 4 schermi, caricare il programma B, immettervi le istruzioni di A presenti sullo schermo, registrare il nuovo B e ripetere l'operazione per il successivo gruppo di istruzioni. La cosa è lunga e non priva di imprevisti (può capitare di non immettere una delle istruzioni di A se non si fa opportunamente scorrere verso l'alto lo schermo, oppure ci si può dimenticare di registrare B prima di caricare A). Il programma binario *DGTSAV* (n. 95029) per l'HP83/HP85 o il *GETSAVEg* (n. 90063B) per l'HP86/HP87 servono per registrare e per leggere dei programmi come dati. Un programma può essere in tal modo caricato in memoria contemporaneamente da un altro, pervenendo in modo semplice alla fusione dei due. Con la prima parte del programma suddetto si registra un programma come una serie di dati, permettendo così di leggerne, modificarne e riscriverne le istruzioni tramite un altro programma; con la seconda parte si carica nella memoria del computer un programma registrato in precedenza come una serie di dati. Questo programma occupa all'incirca 1600 byte di memoria.

```
ELENCO DELLE STRINGHE PRIMA DELL'ORDINAMENTO


TELETYPE EMULATOR (95014)
SORTING GROUPS OF ELEMENTS OF A STRING "SORTB2" (95015)
PROGRAMMER CONTROL OF THE INTERNAL CRT AND KEYBOARD "IPBIN" (95016)
POLAR AND RECTANGULAR COORDINATE CONVERSIONS "RECPOL" (95017)
ENHANCED STRING MANIPULATION "STRNGB" (95018)
STATISTICAL FUNCTIONS ON AN ARRAY "STATBN" (95019)
ARRAY ORGANIZATION AND INITIALIZATION "REDZER" (95020)
ENHANCED STRING INPUT,IMMEDIATE EXECUTE KEYS,AND CURSOR CONTROL 'LINKEY' (9502
CONVERTING ROM STATEMENTS TO MAINFRAME STATEMENTS "UNTRAN" (95022)
COPYING DISCS TO TAPES "COPYT" (95023)
STORING AND RETRIEVING DATA IN STANDARD INTERCHANGE FORMAT "SIF" (95024)
RETRIEVING HP-9845A SAVED PROGRAMS "GET45" (95025)
AUDIBLE BEEP CONTROL "CTRLBP" (95026)
HP-85 GRAPHICS PRINT HEAD CAPABILITIES "PCOL" (95027)
TYPING AIDS "SOFTKEY" (95028)
STORING AND RETRIEVING PROGRAMS AS DATA "DGTSAV" (95029)
GRAPHICS TABLET DISPLAY "TRACK" (95030)
SORT AND FORMS BINARY PROGRAM "SOFORT" (95031)
NUMERIC ARRAY ORGANIZATION AND INITIALIZATION "REDZERg" (9-0056B)
ENHANCED STRING MANIPULATION "STRNGBg" (9-0057B)
MATHEMATICAL FUNCTIONS "MATHBIg" (9-0058B)
GRAPHICS TABLET DISPLAY "TRACKAg" (9-0059B)
GRAPHICS CURSOR "GCURSg" (9-0060B)
ENHANCED STRING INPUT AND CURSOR CONTROL "LINCURg" (9-0061B)
IMMEDIATE EXECUTE KEYS "KEYONg" (9-0062B)
STORING AND RETIREVING PROGRAMS AS DATA "GETSAVEg" (9-0063B)
SORTING GROUPS OF ELEMENTS OF A STRING "SORTB2g" (9-0066B)
ENHANCED CRT, CURSOR AND KEYBOARD CONTROL 'FORMSBg' (9-0067B)
PROGRAMMER CONTROL OF THE INTERNAL CRT AND KEYBOARD "IPBINg" (9-0068B)
LOGICAL INTERCHANGE FORMAT "LIF87" AND "LIFg" (9-0069B)
SORT AND FORMS "SOFORT" (95143)


TEMPO IMPIEGATO:  3.013 SECONDI


ELENCO DELLE STRINGHE DOPO L'ORDINAMENTO


ARRAY ORGANIZATION AND INITIALIZATION "REDZER" (95020)
AUDIBLE BEEP CONTROL "CTRLBP" (95026)
CONVERTING ROM STATEMENTS TO MAINFRAME STATEMENTS "UNTRAN" (95022)
COPYING DISCS TO TAPES "COPYT" (95023)
ENHANCED CRT, CURSOR AND KEYBOARD CONTROL 'FORMSBg' (9-0067B)
ENHANCED STRING INPUT AND CURSOR CONTROL "LINCURg" (9-0061B)
ENHANCED STRING INPUT,IMMEDIATE EXECUTE KEYS,AND CURSOR CONTROL 'LINKEY' (9502
ENHANCED STRING MANIPULATION "STRNGB" (95018)
ENHANCED STRING MANIPULATION "STRNGBg" (9-0057B)
GRAPHICS CURSOR "GCURSg" (9-0060B)
GRAPHICS TABLET DISPLAY "TRACK" (95030)
GRAPHICS TABLET DISPLAY "TRACKAg" (9-0059B)
HP-85 GRAPHICS PRINT HEAD CAPABILITIES "PCOL" (95027)
IMMEDIATE EXECUTE KEYS "KEYONg" (9-0062B)
LOGICAL INTERCHANGE FORMAT "LIF87" AND "LIFg" (9-0069B)
MATHEMATICAL FUNCTIONS "MATHBIg" (9-0058B)
NUMERIC ARRAY ORGANIZATION AND INITIALIZATION "REDZERg" (9-0056B)
POLAR AND RECTANGULAR COORDINATE CONVERSIONS "RECPOL" (95017)
PROGRAMMER CONTROL OF THE INTERNAL CRT AND KEYBOARD "IPBIN" (95016)
PROGRAMMER CONTROL OF THE INTERNAL CRT AND KEYBOARD "IPBINg" (9-0068B)
RETRIEVING HP-9845A SAVED PROGRAMS "GET45" (95025)
SORT AND FORMS "SOFORT" (95143)
SORT AND FORMS BINARY PROGRAM "SOFORT" (95031)
SORTING GROUPS OF ELEMENTS OF A STRING "SORTB2" (95015)
SORTING GROUPS OF ELEMENTS OF A STRING "SORTB2g" (9-0066B)
STATISTICAL FUNCTIONS ON AN ARRAY "STATBN" (95019)
STORING AND RETIREVING PROGRAMS AS DATA "GETSAVEg" (9-0063B,
STORING AND RETRIEVING DATA IN STANDARD INTERCHANGE FORMAT "SIF" (95024)
STORING AND RETRIEVING PROGRAMS AS DATA "DGTSAV" (95029)
TELETYPE EMULATOR (95014)
TYPING AIDS "SOFTKEY" (95028)
```

```
10 ! PROVA
20 DIM A$[8000],B$[80]
30 PRINT "ELENCO DELLE STRINGHE PRIMA DELL'ORDINAMENTO" @ PRINT @ PRINT
40 CLEAR
50 DISP "QUANTE STRINGHE DI 80 CARATTERI VUOI ORDINARE (MASSIMO 100)";
60 INPUT F
70 FOR I=1 TO F
80 INPUT B$@ PRINT B$
90 A$[80*I+1,80*I+80]=B$
100 NEXT I
110 DISP "PREMERE <CONT> PER INIZIARE L'ORDINAMENTO"
120 PAUSE
130 SETTIME 0,0
140 UPCSORT A$,80,1,80
145 A=TIME
150 PRINT @ PRINT @ PRINT "TEMPO IMPIEGATO: ";A;"SECONDI"
160 PRINT @ PRINT @ PRINT "ELENCO DELLE STRINGHE DOPO L'ORDINAMENTO"
170 PRINT @ PRINT
180 FOR I=1 TO F
190 PRINT A$[I*80+1,I*80+80]
200 NEXT I
210 END
```

Uno dei problemi che possono capitare quando si prepara un programma è quello di scrivere il programma con una ROM inutile innestata. Questo programma non può poi essere modificato o eseguito senza la ROM. Se il lavoro è ormai in fase avanzata quando ci si accorge dell'errore, non è più conveniente stare a ribatterlo, anche per la possibilità di errori di battitura, che richiederebbero poi un controllo estremamente accurato del tutto. In tali casi ci si può servire del programma *UNTRAN* (n. 95022) che permette di riportare il programma battuto con la ROM

nel linguaggio del computer senza tale ROM. Occupa 837 byte di memoria.
Vedere un HP85 collegato via satellite con una grossa banca di dati americana è un'esperienza interessante. Il *collegamento* fra due macchine può essere a volte una necessità imprescindibile. Con il programma 95014 l'HP-85 si trasforma in un terminale che si può collegare con altri computer. Da programma si possono regolare il numero di baud, il numero di bit e il numero degli stop. Il programma 95014 fa uso del programma binario *IPBIN* (n. 95016), che dà un completo controllo dello schermo e della tastiera.
Per chi si interessa di statistica c'è *STATBN* (n. 95019), un programma binario contenente cinque calcoli statistici molto usati: deviazione media, deviazione standard, deviazione standard della popolazione, somma dei valori e somma dei quadrati dei valori. I valori vengono calcolati velocemente ed in modo facile. Occupa 872 byte di memoria.
Per restare nel campo dei numeri, *REDZER* (n. 95020) per l'HP83/HP85 e *REDZERg* (n. 90056B) per l'HP86/HP87 permettono di inizializzare a zero tutte le matrici e vettori con un'unica istruzione in tempi molto brevi. Altre otto funzioni matematiche si possono avere per l'HP86/HP87 con *MATHBIg* (n. 90058B), un programma binario di 1569 byte che permette di ottenere le funzioni iperboliche (seno, coseno, tangente, arcotangente, arcoseno e arcocoseno iperbolici), al conversione da numeri decimali ad ottali e viceversa, e i fattoriali con numeri interi o la funzione gamma con i non interi.
Produrre disegni speciali (ad esempio codici a barre) direttamente sulla stampante dell'HP85 (senza passare cioè attraverso la funzione *COPY*) è possibile con *PCOL* (n. 95027), un programma che, con appena 228 byte di occupazione di memoria permette di controllare i punti tracciati dalla testina della stampante termica, in un modo non molto dissimile dalla funzione *BPLOT*.
Uno dei punti deboli del BASIC secondo lo standard ANSI è quello che la virgola è sempre considerata come separatore. Ciò talvolta causa inconvenienti con le istruzioni di input nelle quali si desidera comprendere la virgola (per evitare l'errore in questi casi è sempre necessario racchiudere i dati da immettere fra virgolette). Con il programma *LINKEY* (n. 95021) per l'HP85 e il *LINCURg* (n. 90061B) per l'HP86/HP87 si può immettere qualunque carattere, compresa la virgola e le virgolette, in una variabile "a stringa". Con entrambi i programmi si può poi ottenere uno spostamento del cursore in ogni punto dello schermo. *LINKEY* permette inoltre di assegnare ad ogni tasto della tastiera funzioni speciali. Quest'ultima funzione è svolta per l'HP86/HP87 da un programma a parte *KEYONg* (n. 90062B), che occupa 627 byte di memoria. *LINKEY* occupa 889 byte di memoria, *LINCUR* appena 334. Altri programmi che, con l'HP86/HP87, permettono un ulteriore controllo del cursore e della tastiera sono *FORMSBg* (n. 90067B) e *IPBINg* (n. 90068B).
Per un'ulteriore accentuata manipolazione delle stringhe si tengano presenti i programmi seguenti: *STRNGB* (n. 95018) per l'HP83/HP85 o *STRNGg* (n. 90057B) per l'HP86/HP87 e *SORTB2* (n. 95015) per l'HP83/HP85 o *SORTB2g* (n. 90066B) per l'HP86/HP87.
Otto funzioni speciali di stringa si ottengono con *STRNGB(g)*, fra cui la sottolineatura con l'HP83/HP85 o l'evidenziazione in negativo con l'HP86/HP87, la conversione in minuscole, la ripetizione, l'inversione della stringa, la ricerca e sostituzione.
Quest'ultima funzione è particolarmente interessante, in quanto permette di sostituire parole-chiave, come *ditta, cliente*, ecc. con il nominativo della ditta, del cliente, in testi standard già preparati in precedenza. Dove compariva "Spettabile/ditta/" comparirà "Spettabile Rossi & C. S.p.A.", per esempio.
Con *SORTB2(g)* si ottiene invece la possibilità di riordinare gruppi di elementi di una stringa composta di parti di lunghezza uguale. Una lista di spedizione, per esempio, può essere formata da schede contenenti il nome della ditta, la via, la città e la regione, e queste schede possono essere registrate in un 'unica lunga stringa una dopo l'altra. In tal caso *SORTB2* permette di riordinare le schede per nome, per via, per città, per regione, il tutto molto rapidamente. Occupa 432 byte di memoria.
Il programma che viene presentato fa uso di *SORTB2g* per ordinare alfabeticamente delle stringhe lunghe 80 caratteri considerando tutti i caratteri della stringa. L'istruzione *UPCSORT* fa parte delle istruzioni del programma binario. Si è voluto mettere in ordine alfabetico l'elenco di programmi contenuto nei volumi "uple binary programs set n. 1" e "HP-86/HP-87 bynary programs set n. 1" dai quali s'è tratto spunto per la compilazione dell'articolo. Come si può notare l'ordinamento è rapidissimo, richiedendo appena 3 secondi. ■

ERRATA CORRIGE

Il programma Planel, pubblicato su Riservato Personal di Bit n. 38 a pag. 68, presenta un errore di impaginazione, che ha tagliato la linea 2580. Questa è da leggersi come segue:

2580 R = 0 : IF P<1987 THEN R = = (6 + P − 1986)* 13

Ci scusiamo con i lettori per l'involontario errore.

Facciamo musica con il 64!

Il nuovo VIC 64 ha capacità musicali sorprendenti. Utilizziamole con intelligenza

Il nuovo VIC 64 ha delle caratteristiche molto interessanti e attualmente, tra i personal sul nostro mercato, offre uno tra i migliori rapporti prestazioni-prezzo.
Tra le sue molteplici possibilità c'è anche quella del suono, resa sofisticata dall'uso di un apposito integrato LSI (6581).
In questo articolo vengono presentati tre semplici programmi che permettono appunto di sfruttare le caratteristiche di tale integrato.
Prima di passare alla descrizione di tali programmi, vorremmo riassumere le peculiarità di questo LSI, indicandone le funzioni principali:
- Tre voci indipendenti (peccato non siano quattro), ognuna delle quali copre nove ottave.
- Possibilità di controllo per ognuna dell'ADSR e della forma d'onda, tra quattro disponibili.
- Controllo generale del volume e disponibilità di 22 registri a 8 bit situati dalla locazione 54272 a 54296 per programmare le funzioni.

Per ogni voce si usa un registro da 16 bit (2 di 8) per controllare la frequenza: è perciò possibile suddividere le 9 ottave in 65536 intervalli, ognuno dei quali vale 0,0588 Hz.
Volendo trasformare una frequenza nei valori da inserire nei registri HI e LO, si devono usare queste formule:

HI = INT (Hz/0.0588)/256)
LO = INT((((Hz 0.0588)/256)–
–HI)x256) + 0.5)

dove HI e LO sono i registri che controllano la frequenza della voce (54273 e 54272 per la prima voce).
Per l'ADSR vengono usati da ogni voce due registri da 8 bit ognuno diviso in due registri da 4 bit; per ogni fase dell'inviluppo si hanno 16 possibili valori.
Per l'attacco e per il sostegno vengono usati i 4 bit più significativi delle rispettive locazioni di memoria; invece per il decadimento e per il rilascio, i 4 meno significativi.
Una volta stabiliti i valori per le fasi dell'inviluppo, per ottenere i valori da inserire nei registri a 8 bit si fa: A★16+D, S★16+R, dove le lettere indicano le fasi dell'inviluppo.
Per scegliere la forma dell'onda si usano i 4 bit più significativi del registro di controllo (54276 per la prima voce).
Settando uno alla volta questi bit è possibile avere le seguenti forme d'onda: bit 7 = rumore, bit 6 = onda quadra, bit 5 = sinusoidale, bit 4 = triangolare.
Settando più di uno alla volta questi bit, non si ottiene alcun suono.
Selezionando l'onda quadra o il rumore, è necessario inserire nei registri "PULSE RATE" HI e LO un valore rispettivamente tra 0 e 15, 0 e 255.
I bit 1, 2, 3 non vengono usati.
Il bit 0 controlla l'inizio e la fine dell'inviluppo: quando è portato a 1, l'inviluppo parte; portandolo a 0, l'inviluppo termina producendo la fase di rilascio.
Quindi, per fare terminare un suono, non bisogna azzerare il registro di controllo come dice il manuale, bensì resettare il bit 0, altrimenti si ottiene una brusca interruzione del suono e non avviene la fase di rilascio.
Consigliamo di fissare delle variabili con il valore della forma d'onda desiderata (16, 32, 64, 128), e per iniziare il suono, fare l'OR tra la variabile e 1, come per terminarlo, fare l'AND con 254 (WA OR 1, WA AND 254).
Riassumendo, per ottenere un suono bisogna:
1) Caricare il registro di controllo del volume con un valore tra 0 e 15;
2) Inserire i valori in FREQ HI e LO per ottenere la frequenza desiderata;
3) Caricare i registri AD e SR;
4) Se si usa l'onda quadra o il rumore, caricare i registri PULSE RATE HI e LO;
5) Caricare in una variabile il valore corrispondente alla forma d'onda voluta;
6) Inserire nel registro di controllo la variabile con l'OR 1 (inizia il suono);
7) Fare un ritardo, per esempio, con un ciclo a vuoto;
8) Inserire nel registro di controllo la variabile con l'AND 254 (il suono termina);

Per ultimo, ricordo che in questi 22 registri è possibile solo scrivere e non leggere.
Veniamo ora alla presentazione dei programmi:
Il primo, "Polimus", più che un programma, è una routine che potrete inserire in un qualsiasi vostro programma per "sonorizzarlo".
Polimus legge da un blocco dati innanzitutto i valori da inserire nei registri di controllo e, successivamente, a due a due i valori dell'altezza e durata della nota.
Le linee da 0 a 1796 sono riservate ai dati, sia quelli per Polimus, sia quelli eventualmente usati dal programma che ospita questa routine.
Si è preferito usare la parte iniziale del programma per i dati perchè per poter iniziare a leggerli in un punto qualsiasi del blocco, viene usata una routine che simula l'istruzione BASIC mancante "RESTORE num. di linea", e visto che questa a sua volta è scritta in BASIC è soggetta alla ben nota lentezza di questo linguaggio rispetto al LM.
Questa routine, iniziando dalla linea più bassa, ricerca la linea il cui numero è caricato nella variabile "IL"; quando la trova, carica il suo indirizzo nei puntatori dei "DATA", facendo in modo che i successivi dati letti inizino da quella riga. Più sono le linee che devono essere controllate, maggiore sarà il tempo impiegato da questa routine per svolgere il suo compito. Ovviamente, mettendo i dati dopo il programma, i tempi di ricerca salirebbero ad un livello inaccettabile.
Questa routine è completamente a disposizione, nel caso si volesse usarla, all'interno del programma utente: basta caricare in IL il numero della linea in cui si vuole iniziare a leggere i dati, e richiamarla con un GOSUB 3300.
Per capire il funzionamento di questa routine è utile osservare il suo flow-chart.
Nelle linee 2000 e 2010 vengono calcolati i valori HI e LO per ottenere tutte le note musicali, valori che vengono inseriti nel vettore "Q" a due dimensioni (una per HI e una per LO). Volendo, per esempio, ascoltare la 57° nota, basta inserire nei registri HI e LO del generatore di suoni Q (57,0) per HI e Q (57,1) per LO.
Sul manuale del 64, in appendice, sono riportati i valori delle note e il loro numero progressivo.
Questi valori *danno però* delle note calanti, e perciò ho preferito ricalcolarli con il "La corista" a 440 Hz, in modo che il calcolatore possa suonare anche con altri strumenti musicali.
In queste due linee, per ogni nota viene calcolata la sua frequenza, dopodichè, usando la formula riportata

prima, si calcolano i valori HI e LO che vengono inseriti nel vettore Q.
Si noti che per ottenere la radice dodicesima di due, visto che manca la funzione diretta, si fa 2↑(1/12). Volendo modificare l'intonazione delle note, si deve modificare il valore 440 nella linea 2000.
Per fare questi calcoli, vengono impiegati circa 8 secondi; consiglio pertanto di mettere subito prima di queste linee un'eventuale intestazione, in modo da sfruttare questo tempo morto.
Nella linea 2020 con un ciclo viene caricato gradualmete nel registro del volume (54296) il massimo valore, evitando così un fastidioso "clock" nell'altoparlante.
Nella linea 3000 inizia la routine che trasforma i dati letti in musica.
Nelle prime due linee vengono letti i primi 16 dati che contengono i valori da inserire nei registri ADSR e di controllo nel seguente ordine: LP, HP, AT, SU, (voce 1); LP, HP, AT, SU (voce 2); LP, HP, AT, SU (voce 3); WA, WB, WC, (forma d'onda delle tre voci);
Tempo di esecuzione.
Vengono poi letti due dati alla volta contenenti il numero della nota e il suo valore, generalmente espresso in trentaduesimi. Mettendo al posto del numero della nota uno 0, si ottiene una pausa. Per concludere il brano, basta mettere due 1 al posto della nota e del suo valore.
La routine funziona nel seguente modo.
Ogni voce è controllata da un contatore (A, B, C). Nella linea 3010, "A" viene decrementato, e siccome questa è la prima volta, il suo valore diventerà -1. La condizione "IF" sarà falsa e il salto non viene effettuato.
Nelle linee seguenti si spegne la voce 1 e vengono letti i valori della nota e del tempo che sono inseriti, rispettivamente, nei registri HI e LO tramite il vettore Q e nel contatore "A".
Nella linea 3025 si controlla che il brano non debba terminare. La nuova nota viene fatta poi ripartire. La stessa cosa si ripete per le altre due voci.
Nella linea 3200 viene effettuato un ritardo che scandisce il tempo di ogni trentaduesimo. Si ritorna poi a decrementare il contatore "A": questa volta la condizione "IF" è vera perché il contatore è stato precedentemente caricato e non si avranno le operazioni di lettura. Lo stesso vale per le altre voci. La nota di ogni voce durerà per tanti trentaduesimi quanti ne sono caricati nel relativo contatore.
Per fare un esempio, supponiamo che si debbano leggere i dati delle seguenti note:

Nel contatore "A" viene caricato il Do che vale 2/4 ossia 16/32, in quello "B" la nota Mi che vale essa pure 16/32, in quello "C" il Do che vale 1/4 e cioè 8/32.
Per otto volte i contatori vengono decrementati senza avere altre letture di dati, ma al nono passaggio ci si accorge che il terzo contatore "C" è arrivato a zero: vengono perciò letti altri due dati per questa voce. Per altre otto volte i contatori vengono decrementati, e alla nona tutti e tre contengono 0. Per ogni voce viene letta una coppia di dati (Re, Fa, La) e il tutto si ripete. Dato che le note vengono lette man mano che quelle precedenti si esauriscono, nel blocco dati devono essere ordinate in base alla loro successione ritmica. Si osservi l'esempio di prima: i numeri indicano l'ordine con cui vengono lette le note.
Resta ora da vedere a cosa servono le variabili VP, V6, VX.
"VP" contiene la durata di un ciclo (o di un trentaduesimo), e viene caricata col 16° dato del blocco.
"V6" viene caricata con "VP" ogni volta che si ripete un ciclo.
Se un contatore è superiore allo 0, e pertanto non vengono eseguite le operazioni di lettura, a "V6" viene aggiunta "VX" aumentando così il ritardo di un ciclo; questo serve a compensare la maggior velocità di un passaggio dovuta ad un minor numero di istruzioni BASIC.
Nel caso si volessero usare un numero minore di voci, basta caricare all'inizio del brano una pausa nella o nelle voci che si vogliono escludere, lunga come l'intero brano.
Tradurre un brano musicale in un blocco dati è un lavoro piuttosto uggioso, ed è anche facile commettere errori; per questo esiste un programma che semplifica al massimo tale operazione.
Musicdec permette di scrivere separatamente le voci, specificando il nome della nota, la sua ottava, il suo valore. Durante l'inserzione delle note il programma controlla che non vengano commessi errori, e se ne sperimenta subito l'utilità. È certo possibile durante la trascrizione ascoltare le note inserite, volendo anche voce per voce, ed è possibile correggere eventuali errori o modificare un'intera battuta.
Anche se la funzione principale di questo programma è quella di codificare dei dati per la routine Polimus, si ritiene che esso possa venir utilizzato come strumento didattico per chi deve imparare la musica, o per chi si accinge all'arte della composizione.
Le caratteristiche del programma sono: massimo tre voci, 20 note per battuta, 400 note in totale. Il valore più piccolo usabile è il trentaduesimo.
L'uso del programma è semplicissimo, e tutti i dati richiesti vengono controllati ed eventualmente respinti.
Dato il RUN, il programma mostra l'intestazione: premendo un tasto, si entra nella fase di "input dati". Viene chiesto il numero delle battute e delle voci.
Per la lettura dei dati, il programma usa un'unica routine "RCL" che stampa in un punto dello schermo, ove si voglia, tanti puntini quanti sono i caratteri che si devono leggere; se si deve inserire un numero minore di caratteri bisogna premere RETURN, altrimenti l'accettazione è automatica. In caso di errore si può correggerlo usando il solito tasto.
Successivamente viene chiesto quale frazione c'è in chiave, dopodiché si passa alla fase di inserzione delle note.
Appena scelta la voce, il programma inizia a chiedere le note, battuta per battuta.
Le note possono essere inserite nel formato che segue: prima lettera della nota (fanno eccezione il Sol e il Si per le quali bisogna scrivere le prime due) e numero del-

Facciamo musica con il 64!

l'ottava. Si può anche scrivere il nome per intero della nota con il numero dell'ottava.
Se la nota è alterata, si deve inserire prima del numero d'ottava il Γ o il b.
Per la pausa si deve scrivere una "P".
Dopo la nota o pausa si deve inserire il valore espresso in trentaduesimi.
All'inizio della battuta, premendo RETURN, è possibile cambiare la voce, o premendo "F" andare al menu principale. In questo caso automaticamente viene chiamata la routine che riordina i dati finora inseriti. Durante l'attesa del riordino viene segnalato in alto sullo schermo il numero della battuta in elaborazione.
All'inizio di ogni battuta, con i tasti "+ e −", si può raggiungere un'altra qualsiasi battuta. Per segnalare che si è riscritto sopra una battuta appare dopo la nota una "C". Se invece una battuta è saltata viene automaticamente caricata con una pausa.
La routine che permette l'ascolto dati è simile a Polimus, con la differenza che questa legge da un vettore e non da un blocco dati. Per questo motivo e poiché in queste routine sono presenti più controlli, le note vengono riprodotte più lentamente di Polimus e gli accordi sono leggermente arpeggiati.
Una volta che i programmi sono stati scritti, verificati, e, se necessario, corretti, si possono visualizzare i dati con l'apposita routine.
A questo punto, armati di pazienza, d'altronde ben ripagata, si devono ricopiare tali dati e successivamente riscriverli nel programma che ospita Polimus. Possedendo la stampante consigliamo vivamente di modificare la routine e stampare i dati, evitando di doverli copiare a mano.
Per ultimo due particolari che meritano attenzione.
1) Quando si inseriscono i valori dei registri di controllo, rispondendo con un RETURN, si passa oltre senza modificare il contenuto di quel registro.
2) Se ci si accorge di aver sbagliato una nota nella fase di inserzione si può uscire dalla battuta dando un valore della nota più alto della battuta, simulando così un errore: si potrà allora riscrivere la battuta.
Il terzo programma è un esempio di utilizzo della routine Polimus. Il programma permette di ascoltare quattro motivetti piuttosto noti: potrete rendervi conto delle capacità sonore del 64.
Per chi fosse interessato a ricevere la cassetta contenente i programmi descritti può mettersi in contatto con:

GREMES MIRKO
V.le Mazzini, 62
38015 Lavis (TN)

Note	Pause	Trentaduesimi
		32
		24
		16
		12
		8
		6
		4
		3
		2
		1

Tavola di conversione del valore delle note e pause in trentaduesimi.

```
10 REM------------------------------
11 REM SPAZIO PER I DATI
12 REM------------------------------
1797 REM------------------------------
1798 REM SPAZIO PER INTESTAZIONE PROG.
1799 REM------------------------------
1997 REM------------------------------
1998 REM CARICA IL VET. "Q"
1999 REM------------------------------
2000 DIMQ(90,1):R=2↑(1/12):X=440/(29*256+60):FORT=5TO90:HZ=(R↑T)*16.35
2010 HH=INT(HZ/X):HI%=HH/256:LO%=HH-(HI%*256):Q(T,0)=HI%:Q(T,1)=LO%:NEXT
2020 FORT=0TO15:POKE54296,T:NEXT
2030 GOTO3350
2997 REM------------------------------
2998 REM TRASFORMA I DATI IN MUSICA
2999 REM------------------------------
3000 VX=13:FORT=1TO15STEP7:FORY=1TO5:IFT+Y=4ORT+Y=11ORT+Y=18THENNEXT
3007 READD:POKE54272+T+Y,D:NEXT:NEXT:READWA:READWB:READWC:READVP:A=0:B=0:C=0
3010 V6=VP:A=A-1:IFA>0THENV6=V6+VX:GOTO3040
3020 POKE54276,WAAND254:READD,A:IFD=0THEN3040
3025 IFD=1THEN3210
3030 POKE54272,Q(D,1):POKE54273,Q(D,0):POKE54276,WAOR1
3040 B=B-1:IFB>0THENVV6=V6+VX:GOTO3070
3050 POKE54283,WBAND254:READD,B:IFD=0THEN3070
3055 IFD=1THEN3210
3060 POKE54279,Q(D,1):POKE54280,Q(D,0):POKE54283,WBOR1
3070 C=C-1:IFC>0THENV6=V6+VX:GOTO3200
3080 POKE54290,WCAND254:READD,C:IFD=0THEN3200
3085 IFD=1THEN3210
3090 POKE54286,Q(D,1):POKE54287,Q(D,0):POKE54290,WCOR1
3200 FORT=0TOV6:NEXT:GOTO3010
3210 POKE54276,WAAND254:POKE54283,WBAND254:POKE54290,WCAND254
3220 FORT=0TO100:NEXT:POKE54276,0:POKE54283,0:POKE54290,0:RETURN
3297 REM------------------------------
3298 REM RESTORE NUM. DI LINEA
3299 REM------------------------------
3300 A%=PEEK(43):A1%=PEEK(44)
3305 Q%=A1%*256+A%:B%=PEEK(Q%):B1%=PEEK(Q%+1)
3310 P%=PEEK(Q%+3)*256+PEEK(Q%+2):A%=A%-1:IFA%<0THENA%=255:A1%=A1%=1
3320 IFP%=IL THEN POKE65,A%:POKE66,A1%:RETURN
3330 IFP%>ILTHENRESTORE:RETURN
3340 A%=B%:A1%=B1%:GOTO3305
3350 REM------------------------------
3351 REM INIZIO PROGRAMMA UTENTE
3352 REM------------------------------

READY.
```

```
1 REM ****************
2 REM *              *
3 REM *   POLIMUS    *
4 REM *              *
5 REM *     BY       *
6 REM *              *
7 REM * MIRKO GREMES *
8 REM *              *
9 REM ****************
10 DATA0,15,44,34,111,12,44,32,1,12,44,34,64,64,64,15,0,12,0,12,0
20 DATA12,50,2,47,4,43,4,51,2,52,2,43,4,36,4,60,4,36,4,40
30 DATA4,52,2,60,4,43,4,31,4,52,2,36,4,46,4,60,2,60,10,41
40 DATA4,29,4,48,4,45,4,0,2,28,4,72,2,64,2,74,2,65,2,43
50 DATA4,75,2,66,2,76,2,67,2,31,4,72,2,64,2,74,2,65,2,40
60 DATA4,76,4,67,4,31,4,71,2,62,2,74,4,65,2,43,4,65,2,72
70 DATA12,64,12,36,4,40,4,40,4,50,2,47,4,43,4,51,2,52,2,43
80 DATA4,36,4,60,4,48,4,40,4,52,2,60,4,43,4,31,4,52,2,48
90 DATA4,46,4,60,2,60,12,41,4,29,4,48,4,45,4,40,4,28,4,69
100 DATA2,39,4,27,4,67,2,66,2,38,4,26,4,69,2,72,2,48,4,38
110 DATA4,76,4,38,4,26,4,74,2,72,2,48,4,42,4,69,2,74,12,47
120 DATA4,43,4,43,4,31,4,45,4,33,4,50,2,47,4,35,4,51,2,52
130 DATA2,43,4,36,4,60,4,48,4,40,4,52,2,60,4,43,4,31,4,52
140 DATA2,48,4,43,4,60,2,60,10,41,4,29,4,48,4,45,4,0,2,28
150 DATA4,72,2,64,2,74,2,65,2,43,4,75,2,66,2,76,2,67,2,31
160 DATA4,72,2,64,2,74,2,65,2,40,4,76,4,67,4,31,4,71,2,62
170 DATA2,74,4,65,4,43,4,72,12,64,12,36,4,40,4,43,4,72,2,60
180 DATA2,0,4,74,2,62,2,76,2,64,2,36,4,72,2,60,2,74,2,62
190 DATA2,43,4,76,4,64,4,34,4,72,2,60,2,74,2,62,2,43,4,72
200 DATA2,60,2,76,2,64,2,33,4,72,2,60,2,74,2,62,2,45,4,76
210 DATA4,64,4,32,4,72,2,60,2,74,2,62,2,44,4,72,2,60,2,76
220 DATA2,64,2,31,4,72,2,60,2,74,2,62,2,43,4,76,4,64,4,31
230 DATA4,71,2,59,2,74,4,62,4,43,4,72,16,64,16,36,16,1,1
240 DATA0,0,44,34,0,0,44,18,0,0,0,0,16,16,0,23,62,8,43,16,0,384,55
250 DATA4,57,4,59,4,43,4,60,4,45,4,62,8,47,24,55,8,55,8,64,8,48,24,60,4,62,4,64
260 DATA4,66,4,67,8,47,24,55,8,55,8,60,8,45,24,62,4,60,4,59,4,57,4,59,8,43,24
270 DATA60,4,59,4,57,4,55,4,54,8,50,8,55,4,47,8,57,4,59,4,43,8,55
280 DATA4,57,24,50,8,38,4,48,4,47,4,45,4,62,8,47,16,55,4,57,4,59
290 DATA4,45,8,60,4,62,8,43,8,55,8,47,8,55,8,43,8,64,8,48,24,60,4
300 DATA62,4,64,4,66,4,67,8,47,8,55,8,48,4,47,4,55,8,45,4,43,4,60
310 DATA8,45,16,62,4,60,4,59,4,42,8,57,4,59,8,43,16,60,4,59,4,57
320 DATA4,47,8,55,4,57,8,48,8,59,4,50,8,57,4,55,4,38,8,54,4,55,24
330 DATA43,8,38,8,31,8,1,1
350 DATA0,0,45,28,0,0,45,42,0,0,0,0,16,16,0,50,0,8,0,12,0,372,64
360 DATA2,63,2,64,2,0,12,63,2,64,2,59,2,62,2,60,2
370 DATA57,4,33,2,40,2,0,2,45,2,48,2,0,6,52,2,57,2,59,4
380 DATA28,2,40,2,0,2,44,2,52,2,0,6,56,2,59,2,60,4,33,2
390 DATA40,2,0,2,45,2,52,2,0,6,64,2,63,2,64,2,0,12,63,2
400 DATA64,2,59,2,62,2,60,2,57,4,33,2,40,2,0,2,45,2,48,2
410 DATA0,6,52,2,57,2,59,4,28,2,40,2,0,2,44,2,52,2,0,6
420 DATA60,2,59,2,57,4,33,2,40,2,0,4,45,2,0,6,64,2,63,2
430 DATA64,2,0,12,63,2,64,2,59,2,62,2,60,2,57,4,33,2,40,2
440 DATA0,2,45,2,48,2,0,6,52,2,57,2,59,4,28,2,40,2,0,2
450 DATA44,2,52,2,0,6,56,2,59,2,60,4,33,2,40,2,0,2,45,2
460 DATA52,2,0,6,64,2,63,2,64,2,0,12,63,2,64,2,59,2,62,2
470 DATA60,2,57,4,33,2,40,2,0,2,45,2,48,2,0,6,52,2,57,2
480 DATA59,4,28,2,40,2,0,2,44,2,52,2,0,6,60,2,59,2,57,4
490 DATA33,2,40,2,0,2,45,2,59,2,0,6,60,2,62,2,64,6,36,2
500 DATA43,2,48,2,55,2,0,6,65,2,64,2,62,6,31,2,43,2,47,2
510 DATA53,2,0,6,64,2,62,2,60,6,33,2,40,2,45,2,52,2,0,6
520 DATA62,2,60,2,59,4,28,2,40,2,0,2,52,2,52,2,0,4,64,2
530 DATA0,2,52,2,0,2,64,2,64,2,0,10,76,2,63,2,64,2,63,2
540 DATA64,2,0,12,63,2,64,2,63,2,64,2,63,2,64,2,0,12,63,2
550 DATA64,2,59,2,62,2,60,2,57,4,33,2,40,2,0,2,33,2,48,2
560 DATA0,6,52,2,57,2,59,4,28,2,40,2,0,2,44,2,52,2,0,6
570 DATA56,2,59,2,60,4,33,2,40,2,0,2,45,2,52,2,0,6,64,2
580 DATA63,2,64,2,0,12,63,2,64,2,59,2,62,2,60,2,57,4,33,2
590 DATA40,2,0,2,45,2,48,2,0,6,52,2,57,2,59,4,28,2,40,2
600 DATA0,2,44,2,52,2,0,6,60,2,59,2,57,6,33,6,0,6,0,6,1,1
630 DATA0,0,0,0,0,40,140,0,0,40,140,0,16,16,75
640 DATA0,132,0,12,0,2,53,2,57,2,53,2,60,2,53,2,0,2,65,2
645 DATA41,2,64,1,62,1,45,2,60,1,62,1,41,2,60,1,58,1,48,2
650 DATA57,1,58,1,41,2,57,1,55,1
660 DATA53,2,53,2,52,1,57,2,50,1,48,1,60,2,50,1,48,1,57,2,46,1,45,1,65,2,46,1
670 DATA45,1,60,2,43,1
680 DATA41,2,69,1,72,1,45,2,70,1,72,1,48,2,69,1,72,1,45,2,70,1,72,1,53,2
690 DATA69,1,72,1,48,2,70,1,72,1
700 DATA57,1,65,1,60,1,69,1,58,1,67,1,60,1,69,1,57,1,65,1,60,1,69,1,58,1,67,1
710 DATA60,1,69,1,57,1,65,1,60,1,69,1,58,1,67,1,60,1,69,1
720 DATA53,1,62,1,57,1,65,1,55,1,64,1,57,1,65,1,53,1,62,1,57,1,65,1,55,1,64,1
730 DATA57,1,65,1,53,1,62,1,57,1,65,1,55,1,64,1,57,1,65,1
740 DATA50,1,59,2,53,1,52,1,55,2,53,1,50,1,62,2,53,1,52,1,59,2,53,1,50,1
750 DATA65,2,53,1,52,1,62,2,53,1
760 DATA47,2,67,1,69,1,43,2,67,1,65,1,48,2,64,1,65,1,43,2,64,1,62,1
770 DATA52,2,60,1,62,1,48,2,60,1,58,1
780 DATA53,1,57,2,55,1,53,1,62,1,52,1,60,1,50,1,59,1,52,1,60,1,50,1,59,1
785 DATA48,1,57,1,47,1,55,1,48,1,57,1,47,1,55,1,45,1,53,1
790 DATA43,2,52,1,53,1,48,1,52,1,47,1,50,1,45,1,48,2,47,1,45,1,60,1,43,1
795 DATA59,1,41,1,60,2,43,1,41,1,52,2,40,1
800 DATA38,1,53,2,40,1,38,1,60,2,36,1,43,1,52,2,41,1,40,1,60,2,41,1,43,2,50,2
805 DATA31,2,59,2,24,8,36,8,60,8,1,1,1,1,1,1
1797 REM-----------------------------
1798 REM SPAZIO PER INTESTAZIONE PROG.
1799 REM-----------------------------
1800 PRINT"♥■":POKE53280,246:POKE53281,246
1810 PRINT"▒▒▒▒▒▒▒▒▒▒▒ESEMPIO DI UTILIZZO DEL PROG. ■POLIMUS■"
1830 FORT=0TO39:LL$=LL$+"-":NEXT
1997 REM-----------------------------
1998 REM CARICA IL VET. "Q"
1999 REM-----------------------------
2000 DIMQ(90,1):R=2↑(1/12):X=440/(29*256+60):FORT=5TO90:HZ=(R↑T)*16.35
2010 HH=INT(HZ/X):HI%=HH/256:LO%=HH-(HI%*256):Q(T,0)=HI%:Q(T,1)=LO%:NEXT:VX=13
2020 FORT=0TO15:POKE54296,T:NEXT
2030 GOTO3350
2997 REM-----------------------------
2998 REM TRASFORMA I DATI IN MUSICA
2999 REM-----------------------------
3000 FORT=1TO15STEP7:FORY=1TO5:IFT+Y=4ORT+Y=11ORT+Y=18THENNEXT
3007 READD:POKE54272+T+Y,D:NEXT:NEXT:READWA:READWB:READWC:READVP:A=0:B=0:C=0
3010 V6=VP:A=A-1:IFA>0THENV6=V6+VX:GOTO3040
3020 POKE54276,WAAND254:READD,A:IFD=0THEN3040
3025 IFD=1THEN3210
3030 POKE54272,Q(D,1):POKE54273,Q(D,0):POKE54276,WAOR1
3040 B=B-1:IFB>0THENV6=V6+VX:GOTO3070
3050 POKE54283,WBAND254:READD,B:IFD=0THEN3070
3055 IFD=1THEN3210
3060 POKE54279,Q(D,1):POKE54280,Q(D,0):POKE54283,WBOR1:V6=V6-VX
3070 C=C-1:IFC>0THENV6=V6+VX:GOTO3200
3080 POKE54290,WCAND254:READD,C:IFD=0THEN3200
3085 IFD=1THEN3210
3090 POKE54286,Q(D,1):POKE54287,Q(D,0):POKE54290,WCOR1:V6=V6-VX
3200 FORT=0TOV6:NEXT:GOTO3010
3210 POKE54276,WAAND254:POKE54283,WBAND254:POKE54290,WCAND254
3220 FORT=0TO200:NEXT:POKE54276,0:POKE54283,0:POKE54290,0:RETURN
3297 REM-----------------------------
3298 REM RESTORE NUM. DI LINEA IL
3299 REM-----------------------------
3300 A%=PEEK(43):A1%=PEEK(44)
3305 Q%=A1%*256+A%:B%=PEEK(Q%):B1%=PEEK(Q%+1)
3310 P%=PEEK(Q%+3)*256+PEEK(Q%+2):A%=A%-1:IFA%<0THENA%=255:A1%=A1%=1
3320 IFP%=IL THEN POKE65,A%:POKE66,A1%:RETURN
3330 IFP%>ILTHENRESTORE:RETURN
3340 A%=B%:A1%=B1%:GOTO3305
3350 REM-----------------------------
3351 REM INIZIO PROGRAMMA UTENTE
3352 REM-----------------------------
3360 PRINT"♥"LL$:PRINTTAB(17)"■MENU'■":PRINT"▒"LL$
3370 PRINT"▒■[1]","THE ENTERTAINER (JOPLIN)"
3380 PRINT"▒■[2]","MINUETTO (BACH)"
3390 PRINT"▒■[3]","PERELISA (BEETHOVEN)"
3400 PRINT"▒■[4]","INVENZIONE (BACH)"
3410 PRINT"▒■[5]"," "
3420 PRINT"▒■[6]"," "
3430 PRINT"▒■[7]","FINE"
3440 PRINT"▒▒■■SCEGLI! "
3450 FORT=0TO5:GETT$:NEXT
3460 GETI$:IFI$=""THEN3460
3470 IN=VAL(I$):IFIN<1ORIN>7THEN3460
3480 Q=1234+(80*IN):GOSUB4000
3500 IFIN=1THENRESTORE:GOSUB3000:GOTO3360
3510 IFIN=2THENIL=240:GOSUB3300:GOSUB3000:GOTO3360
3520 IFIN=3THENIL=350:GOSUB3300:GOSUB3000:GOTO3360
3530 IFIN=4THENIL=630:GOSUB3300:GOSUB3000:GOTO3360
3560 IF IN=7THENPRINT"♥":END
3570 GOTO3460
4000 REM-----------------------------
4001 REM REVERSE MARKER
4002 REM-----------------------------
4010 IF Q<2024THENFORT=0TO26:M=PEEK(T+Q):POKE(T+Q),MOR128:NEXT:RETURN

READY.
```

```
1 REM ****************
2 REM *   MUSICDEC   *
3 REM *              *
4 REM *      BY      *
5 REM *              *
6 REM * MIRKO GREMES *
7 REM ****************
8 PRINT"♥■":DIMPT%(2,4):DIMDO%(800)
9 DIMTT(14):POKE53280,6:POKE53281,6:FORT=0TO39:LL$=LL$+"-":NEXT
10 DATA32,24,16,16,12,8,64,48,32,24,36,48,40,48
20 FORT=1TO14:READTT(T):NEXT
30 REM-----------------------------
31 REM INTESTAZIONE
32 REM-----------------------------
33 PRINT"♥";CHR$(14);"▒▒LA FUNZIONE PRINCIPALE DI QUESTO PRO-"
36 PRINT"▒GRAMMA, E' QUELLA DI CODIFICARE UN QUAL-"
40 PRINT"SIASI BRANO MUSICALE IN UN BLOCCO DATI"
45 PRINT"▒NEL FORMATO RICHIESTO DAL PROGRAMMA TL\"
50 PRINT"\♠."
60 PRINT"▒♠ONO PERMESSE AL MASSIMO 400 NOTE,O 70"
70 PRINT"▒BATTUTE."
80 PRINT"▒▒■LE NOTE VANNO INSERITE NEL FORMATO:"
90 PRINT"▒NOME DELLA NOTA E OTTAVA, DURATA ESPRES-"
100 PRINT"▒SA IN TRENTADUESIMI. (T3  8)"
1000 DIMQ(97,1):R=2↑(1/12):X=440/(29*256+60):FORT=12TO97:HZ=(R↑T)*16.35
1010 HH=INT(HZ/X):HI%=HH/256:LO%=HH-(HI%*256):Q(T,0)=HI%:Q(T,1)=LO%:NEXT
1015 POKE54296,15
1020 PRINT"▒■■■■■■■■■■■■■■■■▒REMI UN TASTO!♥■"
1030 GETT$:IFT$=""THEN1030
1040 PRINTCHR$(142)
1885 GOTO6000
1887 REM-----------------------------
1888 REM MENU' ASCOLTO DATI
1889 REM-----------------------------
1900 PRINT"♥"LL$:PRINT"■■■■■■■■■■■▒ASCOLTO DATI■":PRINT"▒"LL$
1910 PRINT"▒▒▒QUALI VOCI VUOI ASCOLTARE? ":WA=0:WB=0:WC=0
1920 FORK=1TONV
1925 RCL=.12031+((K/1E4)*4):GOSUB9200
1930 IFCR>NVTHEN1925
1940 IFCR=3THENWC=DO%(14)
1950 IFCR=2THENWB=DO%(13)
1960 IFCR=1THENWA=DO%(12)
1970 NEXTK
1980 GOTO3010
2997 REM -----------------------------
2998 REM RITARDO
2999 REM -----------------------------
3000 FORT=0TO600:NEXT:RETURN
3007 REM-----------------------------
3008 REM ASCOLTO DATI
3009 REM-----------------------------
3010 CD=0:VX=7:FORT=1TO15STEP7:FORY=1TO5:IFT+Y=4ORT+Y=11ORT+Y=18THENNEXT
3012 D=DO%(CD):POKE54272+T+Y,D:CD=CD+1:NEXT:NEXT:PRINT"▒▒▒"TAB(30)U
3013 CD=CD+3:VP=DO%(CD):CD=CD+1
3014 V6=VP:PRINT"■■■■▒▒"CD:A=A-1:IFA>0THEN3040
3020 POKE54276,WAAND254:D=DO%(CD):CD=CD+1:A=DO%(CD):CD=CD+1:IFD=0THEN3040
3025 IFD=1THEN3210
3030 POKE54272,Q(D,1):POKE54273,Q(D,0):POKE54276,WAOR1:V6=V6-VX
3040 B=B-1:IFB>0THEN3070
3050 POKE54283,WBAND254:D=DO%(CD):CD=CD+1:B=DO%(CD):CD=CD+1:IFD=0THEN3070
3055 IFD=1THEN3210
3060 POKE54279,Q(D,1):POKE54280,Q(D,0):POKE54283,WBOR1:V6=V6-VX
3070 C=C-1:IFC>0THEN3200
3080 POKE54290,WCAND254:D=DO%(CD):CD=CD+1:C=DO%(CD):CD=CD+1:IFD=0THEN3200
3085 IFD=1THEN3210
3090 POKE54286,Q(D,1):POKE54287,Q(D,0):POKE54290,WCOR1:V6=V6-VX
3200 FORT=0TOV6:NEXT:GOTO3014
3210 FORT=0TO999:NEXT:POKE54276,0:POKE54283,0:POKE54290,0:GOTO4100
3999 REM-----------------------------
4097 REM -----------------------------
4098 REM MENU' PRINCIPALE
4099 REM -----------------------------
4100 PRINT"♥"
4120 PRINTLL$;"▒▒■■■■■M E N U'  P R I N C I P A L E":PRINT"▒■"LL$
4130 PRINT"▒■■[1]■","MODIFICA PARAMENTI TIMBRICI"
4140 PRINT"■▒▒▒[2]■","MODIFICA NOTE"
4150 PRINT"■▒▒▒[3]■","ASCOLTO DATI"
4160 PRINT"■▒▒▒[4]■","VISUALIZZA IL BLOCCO DATI"
4165 PRINT"■▒▒▒[5]■","FINE":PRINT"▒▒■■■■■■■■■SCEGLI !"
4170 GOSUB 10000:IFD<1ORD>5THEN4170
4175 Q=1234+120*D:GOSUB9000:GOSUB3000
4177 IFD=1THENRE%=1
4180 ONDGOTO4200,7000,1900,8000,4190
4190 PRINT"♥":END
4197 REM-----------------------------
4198 REM INPUT PARAMENTI TIMBRICI
4199 REM-----------------------------
4200 FORY=1TONV:PRINT"♥"LL$:PRINT"■■■■INPUT PARAMENTI TIMBRICI■":PRINT"▒"LL$
4220 PRINTTAB(20)"VOCE:"Y
4230 PRINT"▒▒▒■HI PULSE (0/15)  LO PULSE (0/255)
4240 RCL=.12052:GOSUB9200
4250 HI=CR:IFHI>15THEN4240
4255 IFCR$=""THEN4260
4257 PT%(Y-1,1)=HI
4260 RCL=.12233:GOSUB9200
4270 LO=CR:IFLO>255THEN4260
4273 IFCR$=""THEN4280
4274 PT%(Y-1,0)=LO
4280 PRINT:PRINT"▒■ATTACK (0/15)    DECAY (0/15)"
4290 RCL=.16052:GOSUB9200
4300 AT=CR:IFAT>15THEN4290
4305 IFCR$=""THEN4340
4310 RCL=.16232:GOSUB9200
4320 DE=CR:IFDE>15THEN4310
4330 PT%(Y-1,2)=AT*16+DE
4340 PRINT:PRINT"▒■SUSTAIN (0/15)   RELEASE (0/15)"
4350 RCL=.20052:GOSUB9200
4355 IFCR$=""THEN 4400
4360 SU=CR:IFSU>15THEN4350
4370 RCL=.20232:GOSUB9200
4380 DE=CR:IFDE>15THEN4370
4390 PT%(Y-1,3)=SU*16+DE
4400 PRINT:PRINT"▒FORMA D'ONDA (1=TRI 2=SIN 3=QUA 4=RUM)"
4410 RCL=.24051:GOSUB9200
4415 IFCR$=""THEN4440
4420 IFCR<1ORCR>4THEN4410
4430 PT%(Y-1,4)=16*(2↑(CR-1))
4440 GOSUB3000:NEXT
4450 PRINT"♥":PRINTLL$:PRINT"■■■■■■▒INPUT VELOCITA'■":PRINT"▒"LL$
4460 PRINT"▒ MIN: 15  MAX: 500"
4470 RCL=.10053:GOSUB 9200
4480 IF CR$=""THEN4500
4490 IFCR<15ORCR>500THEN4470
4495 TP%=CR
4500 IFTP%=0THENTP%=25
4505 GOTO7000
4997 REM-----------------------------
4998 REM INIZIO
4999 REM-----------------------------
5000 PRINT"♥";LL$;"▒"TAB(11)"■TEMPO IN CHIAVE■":PRINT"▒"LL$
5010 PRINT"▒▒■[1]■";"■■4/4 (C)";TAB(24)"■[8]■";"■■3/2"
5020 PRINT"▒▒■[2]■";"■■3/4";TAB(24)"■[9]■";"■■2/2"
5030 PRINT"▒▒■[3]■";"■■2/4";TAB(24)"■[A]■";"■■6/8"
5040 PRINT"▒▒■[4]■";"■■4/8";TAB(24)"■[B]■";"■■9/8"
5050 PRINT"▒▒■[5]■";"■■3/8";TAB(24)"■[C]■";"■■12/8"
5060 PRINT"▒▒■[6]■";"■■2/8";TAB(24)"■[D]■";"■■5/4"
5070 PRINT"▒▒■[7]■";"■■4/2";TAB(24)"■[E]■";"■■6/4":PRINT"▒■■■■■■■■■■■SCEGLI !"
5080 GOSUB10000:TR=TT(D)
5082 IFD<8THENQ=1265+(80*D)
5084 IFD>7THENQ=1288+(80*(D-7))
5090 GOSUB9000:GOSUB3000:GOTO4200
5097 REM-----------------------------
5098 REM RIORDINO DATI
5099 REM-----------------------------
5100 UU=0:FORT=0TO2:FORY=0TO3:DO%(UU)=PT%(T,Y):UU=UU+1:NEXT:NEXT
5120 FORT=0TO2:DO%(UU)=PT%(T,4):UU=UU+1:NEXT
5130 DO%(15)=TP%
5140 IF RE%=1THENRE%=0:RETURN
5170 REM-----------------------------
5180 D2%=0:D3%=0:U=16
5200 CB=1:TA=0:TB=0:TC=0:A%=0:B%=0:C%=0
5210 A1=0:B1=0:C1=0:IFCB>NBTHEN5800
5230 A%=A%-1:IFA%>0THEN5250
5240 A%=VG%(0,CB,1,A1):IFA%=0THENA%=TR
5242 A1=A1+1:TA=TA+A%:IFA1>19THENGOSUB10070:RETURN
5245 DO%(U)=VG%(0,CB,0,A1-1):U=U+1:DO%(U)=A%:U=U+1
5247 IFU>785THENPRINT"TROPPE NOTE!":GOSUB3000:RETURN
5250 IF NV=1ANDD2%=0THENU=U+1:DO%(U)=TR*NB:U=U+2:DO%(U)=TR*NB:U=U+1:D2%=1
```

```
5255 IFNV=1THENTR=TR:TC=TR:GOTO5350
5260 B%=B%-1:IFB%>0THEN5290
5280 B%=VG%(1,CB,1,B1):IFB%=0THENB%=TR
5282 B1=B1+1:TB=TB+B%:IFB1>19THENGOSUB10070:RETURN
5285 DO%(U)=VG%(1,CB,0,B1-1):U=U+1:DO%(U)=B%:U=U+1
5290 IF NV=2ANDD3%=0THENU=U+1:DO%(U)=TR*NP:U=U+1:D3%=1
5295 IFNV=2THENTC=TR:GOTO5350
5300 C%=C%-1:IFC%>0THEN5350
5320 C%=VG%(2,CB,1,C1):IFC%=0THENC%=TR
5322 C1=C1+1:TC=TC+C%:IFC1>19THENGOSUB10070:RETURN
5325 DO%(U)=VG%(2,CB,0,C1-1):U=U+1:DO%(U)=C%:U=U+1
5350 IFTC=TRANDTB=TRANDTA=TRTHENCB=CB+1:TA=0:TB=0:TC=0:PRINT"■■■■"CB:GOTO5210
5360 GOTO5230
5800 DO%(U)=1:U=U+1:DO%(U)=1:GOSUB3000:RETURN
5997 REM-----------------------------
5998 REM INPUT SPECIFICHE
5999 REM-----------------------------
6000 PRINT"■";LL$;"■"TAB(12)"■GETING SPECIFICHE■":PRINT"■"LL$
6010 PRINT"■■■■NUMERO BATTUTE"
6020 RCL=.08182:GOSUB9200
6030 NB=VAL(CR$):IFNB<1ORNB>70THEN6020
6040 PRINT:PRINT"■■■NUMERO VOCI"
6050 RCL=.10181:GOSUB 9200
6060 NV=VAL(CR$):IFNV<1ORNV>3THEN6050
6065 IFDI%=0THENDIMVG%(NV-1,NB,1,20):DI%=1
6075 GOTO5000
6997 REM-----------------------------
6998 REM INPUT NOTE
6999 REM-----------------------------
7000 PRINT"■";LL$;"■"TAB(10)"■INPUT NOTE■":PRINT"■"LL$:PRINT"■■■QUALE VOCE ?"
7015 PRINT"■■■■■■■■■■■■■■■■■F: RITORNO AL MENU'"
7020 RCL=.07181:GOSUB9200:IFCR$="F"THENPRINT"■■ATTENDI!":GOSUB5100:GOTO4100
7030 CV=VAL(CR$):CB=1:IFCV<1ORCV>NVTHEN7020
7033 GOSUB3000
7035 REM-----------------------------
7040 RE%=0:PRINT"■";LL$;"■■■";LL$:IFCB>NBTHENGOTO7000
7050 GOSUB10060
7060 FORW=.05035TO.24035STEP.02
7070 RCL=W:GOSUB9200:WN$=CR$:GOSUB9400:IFER=1THEN7070
7075 IFER=2ORER=4THEN7040
7076 IFER=3THEN7000
7080 RCL=(RCL+.0006)-.00003
7090 GOSUB9200:WT$=CR$:GOSUB9550:IFER=1THEN7090
7095 IFER=2THEN7040
7100 NEXT
7110 FORW=.05215TO.24215STEP.02
7120 RCL=W:GOSUB9200:WN$=CR$:GOSUB9400:IFER=1THEN7120
7130 RCL=(RCL+.0006)-.00003
7140 GOSUB9200:WT$=CR$:GOSUB9550:IFER=1THEN7140
7145 IFER=2THEN7040
7160 NEXT
7170 END
7997 REM-----------------------------
7998 REM VISUALIZZA I DATI
7999 REM-----------------------------
8000 PRINT"■"LL$:CP=0
8005 FORT=0TO9:FORY=0TO9:IFCP>UTHEN8060
8010 PRINTRIGHT$(STR$(DO%(CP)),LEN(STR$(DO%(CP)))-1);",";:CP=CP+1
8020 NEXT:PRINT"■ ■":NEXT
8030 PRINT"PREMI UN TASTO PER VEDERE GLI ALTRI DATI■"
8040 GETT$:IFT$=""THEN8040
8050 PRINT"■"LL$:GOTO 8005
8060 PRINT"■ ":PRINT"■PREMI UN TASTO!"
8070 GETT$:IFT$=""THEN8070
8080 GOTO4100
8997 REM-----------------------------
8998 REM REVERSE MARKER
8999 REM-----------------------------
9000 IFQ<2024THENFORT=0TO26:M=PEEK(T+Q):POKE(T+Q),MOR128:NEXT:RETURN
9097 REM-----------------------------
9098 REM GETING SPECIFICHE
9099 REM-----------------------------
9100 FORT=0TO10:GETT$:NEXT:PO=349
9110 GETD$:IFD$=""THEN9110
9113 IFASC(D$)=13THENRETURN
9115 IFASC(D$)<48ORASC(D$)>57THEN9110
9120 POKE1024+PO,ASC(D$):POKE55296+PO,1:PO=PO+1:DA=DA+VAL(D$):IFPO>350THENRETURN
9130 GOTO9110
9197 REM-----------------------------
9198 REM RCL    RITORNA CR$
9199 REM-----------------------------
9200 RCL$=STR$(RCL):OO=VAL(MID$(RCL$,5,2)):VV=INT(RCL*100):OO$="":VV$="":LP$=""
9210 FORT=1TOOO:OO$=OO$+"■":NEXT:FORT=1TOVV:VV$=VV$+"■":NEXT:LQ$="":CR$=""
9212 NG=VAL(RIGHT$(RCL$,1)):PRINT"■";VV$;OO$;:FORT=1TONG:LP$=LP$+".":LQ$=LQ$+"■
9220 NEXT:PRINT"■";:PRINTLP$;LQ$;:T=0
9230 FORZ=0TO5:GETT$:NEXT
9232 GETD$:IFD$=""THEN9232
9234 IFASC(D$)=13THEN9290
9235 IFD$="P"THENCR$=D$:PRINTD$;:RETURN
9236 IFD$="+"ANDCT=0THENCR$=D$:RETURN
9238 IFD$="-"ANDCT=0THENCR$=D$:RETURN
9245 IFASC(D$)=20ANDT>0THENT=T-1:CR$=LEFT$(CR$,LEN(CR$)-1):PRINT"■.■";:GOTO9280
9260 IFASC(D$)=20THEN9230
9270 CR$=CR$+D$:PRINTD$;:T=T+1:IFT=NGTHEN9290
9280 GOTO9230
9290 CR=VAL(CR$):PRINT"■";:RETURN
9397 REM-----------------------------
9398 REM CONTROLLO NOTE  WN$
9399 REM-----------------------------
9400 NO$=LEFT$(WN$,1):NP$=MID$(WN$,2,1):NO=0:ER=0
9401 IFNO$="P"THENN0=0:OT=1:AL=0:GOTO9520
9402 IFNO$="+"ANDCT=0THENER=4:CB=CB+1:RETURN
9404 IFNO$="-"ANDCT=0THENCB=CB-1:IFCB<1THENCB=1
9405 IFNO$="-"ANDCT=0THENER=4:RETURN
9406 IFNO$=""ANDCT=0THENER=3:RETURN
9410 ER=0:OT=VAL(RIGHT$(WN$,1)):IFOT<1OROT>7THENER=1:RETURN
9412 IFLEN(WN$)<2THENER=1:RETURN
9415 AL$=MID$(WN$,(LEN(WN$)-1),1):AL=0:IFAL$="#"THENAL=1
9420 IFAL$="B"THENAL=-1
9440 IFNO$="D"THENNO=12
9450 IFNO$="R"THENNO=14
9460 IFNO$="M"THENNO=16
9470 IFNO$="F"THENNO=17
9480 IFNO$="L"THENNO=21
9490 IFNO$="S"ANDNP$="O"THENNO=19
9500 IFNO$="S"ANDNP$="I"THENNO=23
9510 IFNO=0THENER=1:RETURN
9520 IF CT=0THENGOSUB9600
9530 VG%(CV-1,CB,0,CN)=NO+((OT-1)*12)+AL:RETURN
9547 REM-----------------------------
9548 REM CONTROLLO TEMPO  INCREM CONTAT
9549 REM-----------------------------
9550 ER=0:WT=VAL(WT$):IFWT<1ORWT>99THENER=1:RETURN
9560 CT=CT+WT:IFCT>TRTHENGOSUB9600:CT=0:CN=0:ER=2:RETURN
9565 VG%(CV-1,CB,1,CN)=WT
9570 IFCT=TRTHENCT=0:CB=CB+1:CN=0:ER=2:RETURN
9580 CN=CN+1:IFCN>20THENGOSUB9600:CT=0:ER=2:RETURN
9590 RETURN
9597 REM-----------------------------
9598 REM CANCELLA BATTUTA IN CORSO
9599 REM-----------------------------
9600 IF VG%(CV-1,CB,1,CN)=0THENRETURN
9630 FORT=0TO1:FORY=0TO20:VG%(CV-1,CB,T,Y)=0:NEXT:NEXT:PRINT" C":RETURN
9997 REM-----------------------------
9998 REM GETING
9999 REM-----------------------------
10000 FORT=0TO10:GETT$:NEXT
10010 GETD$:IFD$=""THEN10010
10020 D=ASC(D$):IFD<48ORD>69THEN10000
10030 D=D-48:IFD>9THEND=D-7
10040 RETURN
10057 REM-----------------------------
10058 REM INDICATORE VOC BAT
10059 REM-----------------------------
10060 PRINT"■■■■■NUMERO VOCE:■";CV;"■■■■NUMERO BATTUTA:■ ";CB:RETURN
10067 REM-----------------------------
10068 REM SEGNALA L'ERRORE
10069 REM-----------------------------
10070 PRINT"■ERRORE NELLA BATTUTA"CB:GOSUB3000:GOSUB3000:RETURN

READY.
```

Per chi volesse approfondire la teoria musicale applicata ai personal, in questo numero di **Bit** appare anche un articolo dello stesso autore che amplia e approfondisce l'argomento.

Variabili usate:

Prog. Polimus
Q(90,1): Contiene i valori HI (Q(X,0)) e LO (Q(X,1)) delle note musicali.
VX: Valore di compensazione (vedi articolo).
WA, WB, WC: Contengono il codice della forma d'onda selezionata.
V6: Contiene il valore della durata di un ciclo (un trentaduesimo) da cui, se necessario, viene tolto il valore di compensazione contenuto in VX.
VP: Valore di un ciclo.
A, B, C: Contatori delle tre voci.
A%, A1%, B%, B1%, Q%: Variabili di comando usate nella routine "RESTORE num. di linea".
R, HZ, T, HI%, LO%, HH, Y, D: Variabili di comodo usate nel programma.

Prog. Esempio
Le stesse usate da Polimus più:
LL$: Contiene 40 "—".
T$, IN, Q, I: Variabili di comodo.

Prog. Musicdec
VG% (NV, NB, N/T, NN): Vettore a quattro dimensioni: numero della voce, numero della battuta, codice nota e tempo nota, numero della nota all'interno della battuta.
CV, CN, CB, CT: Contatori voce, nota, battuta, tempo.
NB, NV: Numero delle battute e delle voci che compongono il brano.
LL$: Contiene 40 "—".
ER: Contiene il codice errore.
RCL: Variabile che stabilisce quanti caratteri devono essere letti, e in quale zona del video devono essere visualizzati. RCL deve essere caricata prima di chiamare l'omonima routine.
CR$, CR: Variabili restituite dalla routine "RCL"; contengono i dati letti.
WN$: Contiene il nome della nota in esame.
WT: Contiene il tempo della nota in esame.
DO%: Vettore contenente i dati ordinati.
PT%: Vettore a tre dimensioni che contiene i valori per i registri di controllo.

REMark "Musicdec"

8-20 *Vengono dimensionati i vettori, vengono selezionati i colori del bordo e dello sfondo, LL$, viene caricata con 40 "-", il vettore T viene caricato con il numero di trentaduesimi di ogni frazione musicale.*
30-100 *Intestazione.*
1000-1400 *Sono calcolati i valori HI e LO per le note.*
1900-1980 *Menù ascolto dati; viene chiesto quali voci si vogliono ascoltare.*
3000-3210 *È una routine simile a Polimus, con la differenza che i dati vengono letti da un vettore e non da un blocco dati.*
4100-4180 *Menù principale: viene richiesto quale opzione del programma si vuole usare.*
4200-4505 *Vengono chiesti i valori da inserire nei registri di controllo e la velocità di esecuzione.*
5000-5090 *Viene chiesto quale frazione c'è in chiave del brano che si vuole trascrivere.*
5100-5800 *Questa routine legge i dati contenuti nel vettore VG%, dove sono ordinati per voce-battuta, e li trasforma nel formato richiesto dal programma Polimus.*
6000-6075 *Vengono richiesti il numero delle battute, e delle voci che si vogliono trascrivere.*
7000-7160 *Vengono richieste le note e i relativi valori.*
8000-8080 *Visualizza i dati ordinati, in modo che possano essere trascritti.*
9000 *Questa linea trasforma in negativo il contenuto della linea video che inizia nella posizione scritta in "Q".*
9100-9130 *Legge i dati dalla tastiera.*
9200-9290 *Questa routine legge i dati dalla tastiera e li visualizza in un punto dello schermo determinato dal contenuto della variabile "RCL" che deve essere caricata prima di richiamare la routine. Questa variabile contiene anche il numero di caratteri che devono essere letti; se i caratteri che si vogliono inserire sono di meno di quelli richiesti si deve premere RETURN.*
9400-9530 *Dopo che le note sono state lette vengono passate a questa routine che le trasforma nel numero corrispondente.*
9548-9590 *Controlla il tempo delle battute, se non ci sono errori incrementa i contatori "note e battute".*
9600-9630 *Se si sovrascrive in una battuta già scritta, viene richiamata questa routine che cancella quello precedentemente scritto.*
10000-10040 *Legge dalla tastiera la scelta del menù.*
10060 *Nella fase di inserimento delle note, mostra il numero della voce e della battuta.*
10070 *Segnala dove è presente un eventuale errore.*

Le derivate simboliche in Pascal

"È indegno di un matematico perdere il suo tempo facendo calcoli"
Gottfried Wilhelm Leibnitz

**di E. Silvestri
e P. Bresciani**

Con tale citazione mi consolavo quando sbagliavo a fare i calcoli delle derivate negli esercizi di analisi. Ma spesso non basta consolarsi, bisogna fare i calcoli giusti! Ed è così che mi è venuta l'idea di insegnare all'Apple a fare le derivate per me. L'algoritmo risolutivo è ricorsivo, a causa della natura stessa del problema, pertanto la scelta del linguaggio è ricaduta su un linguaggio che permette la ricorsione: il Pascal.
Perché il Pascal? Certamente il Lisp sarebbe stato più adatto a tale compito, ma il Pascal presenta, una volta superate le difficoltà iniziali, molti vantaggi, tra cui:
— È disponibile su moltissime macchine tra cui molti personal come appunto l'Apple. (Sul numero di Marzo '82 di Electronic Design è indicato come quello di gran lunga più diffuso sui nuovi sistemi operativi: il 70% supporta il Pascal, il 40% il Fortran, il 37% il BASIC).
— È un linguaggio tutto sommato relativamente semplice, molto chiaro e molto facile da documentare. (Non per niente Wirth lo ha concepito come linguaggio didattico!)
Prima di cominciare a parlare del programma un'ultima nota: il Pascal standard non dispone del tipo string, e ciò può rendere i programmi abbastanza pesanti poiché devono contenere anche tutte le primitive di gestione di stringhe. A questo fortunatamente ovvìa la versione UCSD (cioè quella dell'Apple) mettendo a disposizione il tipo "string" con tutte le procedure e funzioni necessarie. Questo ha sicuramente alleggerito il lavoro, ma ha notevolmente compromesso la portabilità del programma su altre macchine.

Il programma

Ora diamo un'occhiata alla struttura del programma. Esso, in pratica, è costituito da quattro procedure base che sono *carica, deriva, semplifica* e *calcolo*.
Ne darò qui una breve e sommaria descrizione; chi vuole approfondire può guardare direttamente il listing con i commenti, che dovrebbero essere sufficienti per una buona comprensione del programma.
— *"Carica"* ha il compito di verificare la correttezza dell'espressione che le viene passata come parametro (cosiddetto parsing) e contemporaneamente di metterla in un albero costituito da record varianti di tipo "grop". Le espressioni accettate sono tutte e solo quelle che si possono costruire con le carte sintattiche di figura 1. Non mi fermo qui a parlare delle carte sintattiche: sono talmente chiare che non c'è bisogno di alcuna spiegazione. Vale piuttosto la pena di fare un appunto riguardo l'esponenziazione: visto che non vale l'associativa (cioè (x^y)^z <> x^(y^z)) il programma non accetterà due operandi di esponenziazione consecutivi (cioè x^y^z), ma essi dovranno essere separati da una coppia di parentesi (cioè (x^y)^z).
Faccio qui notare che, poiché il programma accetta in *INPUT* una stringa conforme alla sintassi ed, eventualmente, un *tappo* (che non è obbligatorio in quanto viene comunque concatenato dal programma), accade che tutto ciò che segue tale *tappo* viene ignorato dalla *carica*. Perciò nella stringa x^y^z il secondo ^ viene interpretato come *tappo* e ciò che segue, cioè z, ignorato. Se l'espressione non è conforme alle carte sintattiche il programma abortisce posizionando un ^ sotto al punto dove ha trovato l'errore. Ho scelto questa soluzione drastica perché *carica* a questo punto ha già allocato la memoria per rappresentare la stringa nell'albero, e questa memoria non è recuperabile dal programma (almeno così come è adesso). Ciò potrebbe, se ripetuto, ridurre troppo la memoria disponibile e causare un errore in fase d'esecuzione.
A questo punto la parte più difficile del programma è fatta, poiché l'albero così ottenuto è la struttura dati su cui operano le altre procedure.
— *"Deriva"* interviene una volta caricato l'albero e lo visita in modo ricorsivo facendone la derivata. Quest'ultima viene messa nella stringa *out* che è esterna, cioè comune a tutto il programma. E qui c'è un'evidente limitazione: la derivata non può essere più lunga di 255 caratteri, cioè la lunghezza massima delle stringhe permessa dal Pascal UCSD. Per non appesantire ulteriormente il programma non ho messo nessun controllo sulla lunghezza della stringa, e così si avrà un errore in fase d'esecuzione se ciò dovesse capitare. Normalmente questa non dovrebbe essere una cosa grave, ma per determinate applicazioni che richiedono le derivate di ordine maggiore (per esempio i polinomi di Taylor) oppure per funzioni molto lunghe e complicate può risultare limitativo. Una soluzione a ciò potrebbe essere quella di modificare la procedura in modo da farle mettere la derivata non in una stringa ma direttamente in un altro albero. In questo caso l'unico limite sarebbe dato dalla memoria utente a disposizione.
— *"Semplifica"* serve ad ovviare in parte a questo inconveniente e per non operare su espressioni ridondanti che sarebbero poco leggibili e che rallenterebbero il calcolo. Il suo compito è appunto quello di togliere le parti non significative e questo lo fa non sulla stringa ma direttamente sull'albero "potando" i rami e le foglie che non danno contributo all'espressione. Se, per esempio, la funzione che vogliamo studiare è y = 6★x la sua derivata, calcolata in modo meccanico, sarà 0 ★ x + 6 ★ 1, che è ovviamente uguale a 6. Nell'albero verrà rappresentata come in figura 2.
Semplifica comincerà a sostituire tutto il sottoalbero di sinistra con uno zero, poi sostituirà quello di destra con un sei ed infine calcolerà 0 + 6 = 6. Un difetto è che non recupera la memoria occupata dagli elementi "potati" all'albero. Per ovviare a questo fatto sull'Apple bisognerebbe organizzarsi una lista libera, visto che manca la procedura standard release. Un'altra manchevolezza è che non toglie le parentesi ridondanti. Ma questo non si può fare sull'albero, poiché esse vengono inserite solamente in fase di scrittura. Comunque esse non interferiscono né nella valutazione di una funzione né nel calcolo delle derivate, e quindi non costituiscono una limitazione per le applicazioni pratiche, anche se ovviamente limitano la leggibilità dell'espressione. Comunque lo scopo di questo programma è *quello* di fare le derivate e non quello di essere un semplificatore di funzioni. D'altra parte un buon semplificatore meriterebbe un discorso a parte, che potrebbe, se fatto bene, essere ben più complicato della derivazione qui presentata!

In effetti, scrivere una procedura in grado di semplificare una qualsiasi espressione è una impresa impossibile, giacché, citando testualmente "Le Scienze" n. 162 pag. 87: "... non esiste un algoritmo che da solo possa ridurre ogni espressione alla sua forma più semplice." E: "Tuttavia (ivi pag. 92) non vi è accordo su quale sia la forma più semplice possibile di una espressione".
— *"Calcolo"* è infine una funzione che ha il compito di calcolare, dato un numero reale x, il corrispondente valore della funzione rappresentata nell'albero la cui radice le viene passata. Naturalmente ciò non è sempre possibile, vuoi perché il valore è troppo grande e si ha overflow, vuoi perché in quel punto la funzione non è definita (per esempio il logaritmo di un numero negativo o una divisione per zero). Per ovviare a ciò ho introdotto una variabile globale di tipo errore che viene settata a seconda del tipo di errore incontrato.
A questo punto vorrei fare una nota: queste procedure si potrebbero anche fare, e forse in maniera più semplice, definendo "grop" non come record variante ma semplicemente come stringa. Ma i difetti di questa soluzione sono evidenti:
— Il calcolo verrebbe molto rallentato, il che sarebbe una grossa limitazione per le applicazioni pratiche dove sono richieste frequenti valutazioni della funzione e delle sue derivate.
— Limiterebbe ancora di più la portabilità del programma, per via del fatto che le stringhe non sono Pascal standard.
— Andrebbe contro lo spirito del Pascal, che prevede appunto i tipi enumerativi e i record varianti per ottimizzare i tempi d'esecuzione e l'occupazione di memoria.

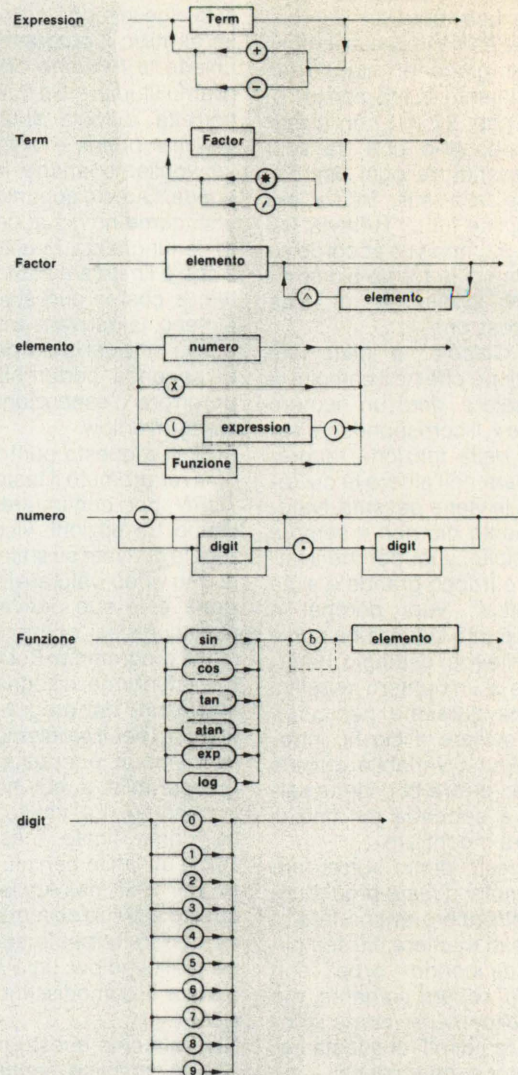
Uso del programma

Le funzioni che svolge questo programma sono queste:
— calcolo e visualizzazione (su video o stampante) della derivata prima e seconda
— calcolo di singoli valori della funzione o delle sue derivate.
L'uso dovrebbe essere abbastanza comodo, in quanto si svolge in modo colloquiale: all'inizio il programma ci chiede la funzione che vogliamo studiare. Se questa è corretta calcola subito la derivata prima, e ci chiede se vogliamo anche la seconda. Questo appunto perché, come ho già accennato, la lunghezza di una funzione è limitata a 255 caratteri, e così si può studiare, almeno, la derivata prima di quelle funzioni la cui derivata seconda porterebbe ad un errore d'esecuzione per string overflow.
Arrivati a questo punto, dopo aver premuto il tasto *RETURN* per continuare, abbiamo tre opzioni: visualizzare le derivate su stampante o su video, valutare la funzione e le sue derivate in punti qualsiasi, oppure uscire dal programma. Per cambiare funzione bisogna prima uscire dal programma, appunto per il problema della memoria occupata non recuperabile a cui ho già accennato. La valutazione ha come limite massimo *10E5*, un limite ben più basso dei limiti dell'Apple, ma questo era l'unico modo per fare un trattamento semplice dell'overflow (in Pascal manca il comodissimo "on error").
Naturalmente questo non è che un semplice esempio di applicazione delle procedure base, e se ne potrebbero fare tanti altri. Quello più immediato è senza dubbio un programma di studio di funzioni che non si limiti a considerare la sola funzione, ma anche le sue derivate. Ma esso non esaurisce certamente tutte le possibilità, poiché moltissimi metodi numerici usano le derivate (per esempio la soluzione di equazioni differenziali col metodo dei polinomi di Taylor oppure la ricerca degli zeri di una funzione col metodo di Newton). Inoltre, con semplici modifiche e senza cambiare nulla alla struttura delle procedure, è possibile ottenere anche le derivate parziali, che aprirebbero la via ad applicazioni ancora più interessanti.
Un'altro uso di questo programma può essere quello didattico: infatti *carica* rappresenta un valido esempio di analizzatore di funzioni (parser) e semplifica un semplice esempio di semplificatore. Inoltre il programma è un valido esempio di come usare alcune caratteristiche molto interessanti del Pascal, quali la ricorsione, i record varianti, le strutture dati dinamiche e la possibilità di definizione di tipi interni per enumerazione, che normalmente non vengono quasi mai illustrate a sufficienza.
Per concludere vorrei scusarmi umilmente coi principianti e i *"BASIC*-glotti" (cioè quelli che parlano solo il BASIC), che non troveranno del tutto semplice capire il funzionamento del programma. Ma i primi possono sempre progredire e i secondi converranno che il BASIC, pur essendo un linguaggio abbastanza comodo, proprio non si adattava al problema. ■

Bibliografia

Sul Pascal standard: Wirth, Pascal user manual and report, Spriger Verlag;
Grogono, Programming in Pascal, Addison Wesley.
Sul Pascal UCSD: il manuale sul Pascal dell'Apple.
Sulla ricorsione, strutture dati dinamiche ecc.: Wirth, Algorithms + Data Structures = Programs, Prentice Hall
Knuth, The Art of Computer Programming, vol. III, Addison Wesley.
Sulle derivate: Qualsiasi testo di analisi (anche se di solito viene enfatizzata più la parte analitica di quella algebrica).
Sull'algebra al calcolatore vedi, come già citato l'articolo su: "Le scienze" n. 162.


```
                         NOME DEL FILE: MAIN.TEXT


(*************************************************************************)
(*#######################################################################*)
(*************************************************************************)

(*                      'DERIVATE  ALGEBRICHE'
                 BY ENRICO SILVESTRI & PAOLO BRESCIANI                  *)


(*$S++*)

PROGRAM DER;
USES TRANSCEND;

(**********************         DICIARATIVE         **********************)

 CONST DIM  =255;     (* DIMENSIONE DELLA STRINGA *)
       OVER =10E5;    (* MASSIMO NUMERO TRATTABILE *)
       UNDER=10E-5;   (* IL RECIPROCO DI OVER *)


 TYPE STRI=STRING[DIM];
      TIPOERRORE=(NESSUNO,DIVBYZER,EXPNEG,LOGNEG,OVERFLOW);
                (* TIPI DI ERRORI TRATTABILI *)
      TIPONODO=(OPER,FUNZ,NUM,VR);
                (* TAG DI CONTROLLO PER COSTRUIRE I NODI A RECORDS VARIANTI *)
      OP=(PIU,MENO,PER,DIVI,ELEV);
      FZ=(SENO,COSENO,LGN,EXPON,TAN,ATANG);
      PTR=^GROP;      (* GROP SIGNIFICA NODO *)
      GROP=RECORD CASE TIPO:TIPONODO OF
                   OPER:(OPERAZIONE:OP;SX,DX:PTR);
                   FUNZ:(FUNZIONE:FZ;ARG:PTR);
                   NUM:(VALORE:REAL);
                  END;


VAR   ROOT0,ROOT1,ROOT2:PTR;  (* PUNTANO RISPETTIVAMENTE ALL'ALBERO DELLA
                                 FUNZIONE E DELLE SUE DERIVATE I E II  *)
      CH1,CH,CR(* CARRIAGE RETURN *),BELL:CHAR;
      EN,  (* STRINGA DI INPUT *)
      OUT, (* STRINGA DI SERVIZIO *)
      OUT1,OUT2:STRI;  (* STRINGHE PER LE DERIVATE *)
      ERRORE:TIPOERRORE;
      STAMPANTE:TEXT;
      M:1..DIM;        (* INDICE AL CORRENTE CARATTERE PER IL CARICAMENTO
                          NELL'ALBERO (SERVE ANCHE PER POSIZIONARE LA
                          SEGNALAZIONE D'ERRORE *)


(******************          FINE DICHIARATIVE          *****************)

PROCEDURE ERROR;  (* SE NON RICONOSCE UNA FUNZIONE *)
  BEGIN
    WRITE(BELL);
    WRITELN(CR,CR,'ERRORE: PREMI U E RISCRIVI L''ESPRESSIONE');
    WRITELN(CR,'FUNZIONE NON CONFORME');
    WRITELN(CR,EN,CR,'^':M);
    EXIT(PROGRAM)
  END;


PROCEDURE ERROR1; (* SE TROVA UN ERRORE SULLE PARENTESI *)
  BEGIN
    WRITE(BELL);
    WRITELN(CR,CR,'ERRORE: PREMI U E RISCRIVI L''ESPRESSIONE');
    WRITELN(CR,'ASPETTASI PARENTESI: )');
    WRITELN(CR,EN,CR,'^':M-2);
    EXIT(PROGRAM)
  END;

(****************          OPZIONI DI INCLUDE FILE          ***************)

(*$I #4:CARICA *)
(*$I #4:DERIVA *)
(*$I #4:CALCOLO *)
(*$I #4:SEMPLIFICA *)

(*************************************************************************)

 FUNCTION DERIVABENE(PRIMITIVA:PTR): (* PUNTA ALL'ALBERO DELLA PRIMITIVA *)
                   PTR;              (* PUNTA ALL'ALBERO DELLA DERIVATA  *)

      (*  NOTA: LA DERIVATA VIENE MESSA IN OUT CHE E' ESTERNA   *)
```

```
VAR P:PTR;
BEGIN
  OUT:='';
  DERIVA(PRIMITIVA);  (* LA DERIVATA NON SEMPLIFICATA E'IN OUT *)
  OUT:=CONCAT(OUT,'# ');  (* METTE IL TAPPO *)
  P:=CARICA(OUT);       (* P PUNTA ALL'ALBERO DELLA DERIVATA *)
  SEMPLIFICA(P);        (* P PUNTA ALL'ALBERO SEMPLIFICATO *)
  OUT:='';
  SCRIVI(P);            (* SCRIVE SU OUT L'ALBERO SEMLIFICATO *)
  OUT:=CONCAT(OUT,'& ');  (* METTE IL TAPPO A OUT *)
  DERIVABENE:=P
END;

PROCEDURE DISPLAY(STRINGA:STRI); (* SCRIVE UNA STRINGA SPEZZANDOLA *)
 CONST ACAPO=40;    (* LUNGHEZZA DI UNA SINGOLA RIGA DI TESTO *)
 VAR I:INTEGER;
 BEGIN
   FOR I:=1 TO LENGTH(STRINGA) DO
     BEGIN
       IF (I MOD ACAPO) = 0 THEN WRITELN;
       IF NOT (STRINGA[I] IN ['@','$','&','#']) (* TERMINATORI *)
         THEN WRITE(STRINGA[I])
     END;  (* FOR *)
   WRITELN(CR,CR)
 END;  (* DISPLAY *)

PROCEDURE INIZIA;  (* ACQUISISCE LA FUNZIONE DA DERIVARE E NE FA LE DERIVATE *)
 BEGIN
  OUT:='';
   PAGE(OUTPUT);
   WRITELN(CR);
  WRITELN('    DAMMI LA FUNZIONE',CR);
   WRITE('Y= ');
   READLN(EN);              (* IN EN C'E' LA FUNZIONE *)

   EN:=CONCAT(EN,'@ ');     (* METTE IL TAPPO *)
   ROOT0:=CARICA(EN);       (* ROOT0 PUNTA ALLA FUNZ *)
   OUT1:='';
   WRITELN('LA FUNZIONE E'' CORRETTA');

   SEMPLIFICA(ROOT0);  (* SEMPLIFICA SOLO L'ALBERO E LASCIA
                          INALTERATA LA STRINGA EN *)

   WRITELN(CR,'CONTROLLO SULLA FUNZIONE :');
   SCRIVI(ROOT0);     (* IN OUT C'E' LA FUNZIONE EVENTUALMENTE SEMPLIFICATA *)
   WRITELN(CR,'Y= ',OUT);(* RISCRIVE LA FUNZIONE EVENTUALMENTE SEMPLIFCATA*)
   WRITELN;
   OUT:='';
   ROOT1:=DERIVABENE(ROOT0);  (* LA DERIVATA E' IN OUT *)
   OUT1:=OUT;
   WRITELN('DERIVATA I : ');
   DISPLAY(OUT1);

   WRITELN('VUOI ANCHE LA DERIVATA SECONDA ? (S/N)');
   REPEAT
     READ(KEYBOARD,CH1)
   UNTIL CH1 IN ['S','N'];
   IF CH1='N' THEN ROOT2:=NIL
            ELSE BEGIN
                   ROOT2:=DERIVABENE(ROOT1);
                   OUT2:=OUT;
                   WRITELN('DERIVATA II:');
                   DISPLAY(OUT2)
                 END;  (* ELSE *)

   WRITELN(CR,CR,'(RET) PER CONTINUARE');
   READLN;
 END;

PROCEDURE CONTINUA;  (* GESTISCE L'INTERFACCIA UTENTE *)

 PROCEDURE CALC;  (* CALCOLA SINGOLI VALORI DELLA FUNZIONE
                     O DELLE SUE DERIVATE *)
  VAR YVAL,XVAL:REAL;
      CH,ST:CHAR;
      RADIX:PTR;
  BEGIN
    PAGE(OUTPUT);
    GOTOXY(2,5);
    WRITELN('CHE COSA VUOI CALCOLARE ?');
    WRITELN('1 - LA FUNZ');
    WRITELN('2 - LA DERIV 1');
    WRITELN('3 - LA DERIV 2');
    REPEAT
      READ(KEYBOARD,CH)
    UNTIL CH IN ['1'..'3'];
    CASE CH OF
      '1' : RADIX:=ROOT0;
      '2' : RADIX:=ROOT1;
      '3' : IF ROOT2<>NIL THEN RADIX:=ROOT2
                          ELSE BEGIN
                                 WRITELN(BELL,'F" NON DISPONIBILE !');
                                 READLN;
                                 EXIT(CALC)
                               END
    END;  (*  DEL CASE  *)

    REPEAT
      WRITE(CR,CR,CR,'VALORE X DA CALCOLARE : ');
      READ(XVAL);
      WRITELN;
      ERRORE:=NESSUNO;
      YVAL:=CALCOLO(RADIX,XVAL);
      CASE ERRORE OF
        NESSUNO:WRITELN('X= ',XVAL:6:17,'-> ',YVAL);
        OVERFLOW:WRITELN('CALCOLO ABORTITO CAUSA OVERFLOW');
        DIVBYZER:WRITELN('CALCOLO ABORTITO CAUSA',CR,'DIVISIONE PER ZERO');
        EXPNEG  :WRITELN('CALCOLO ABORTITO CAUSA',CR,
                         'ELEVAZIONE DI UN NUMERO NEGATIVO');
        LOGNEG  :WRITELN('CALCOLO INTERROTTO CAUSA',CR,
                         'LOGARITMO DI UN NUMERO NEGATIVO')
      END; (* CASE *)
      WRITE(CR,'ANCORA ?',CR);
      READ(CH);  (* LEGGE IL BLANK DEL RETURN *)
      READ(KEYBOARD,CH)  (* LEGGE LA RISPOSTA *)
    UNTIL CH='N';
  END; (* DI CALC *)

 PROCEDURE FAIVEDERE;  (* VISUALIZZA LA FUNZIONE E LE SUE DERIVATE SU
                          VIDEO O STAMPANTE *)
  VAR ST:CHAR;
  BEGIN
    PAGE(OUTPUT);
    GOTOXY(0,6);
    WRITELN(CR,'SE VUOI L''OUTPUT SU STAMPANTE PREMI S');
    WRITELN('QUALSIASI ALTRO TASTO ALTRIMENTI');
    READ(KEYBOARD,ST);
    IF ST='S' THEN BEGIN
                    (*$I-*)
                    REWRITE(STAMPANTE,'PRINTER:');
                    IF IORESULT<>0
                      THEN BEGIN
                             WRITELN(CR,CR,CR);
                             WRITELN('STAMPANTE NON E'' COLLEGATA !');
                             WRITELN(CR,'(RET) PER CONTINUARE');
```

Nota: *ovviamente '–' equivale a '– 1'*

Nota: *'ƀ' può essere omesso solo se 'elemento' comincia con '('*

Nota: *gli unici ƀ richiesti sono quelli segnati. Tuttavia se ne possono inserire a piacere, purché non all'interno di reserved words.*

Figura 1 - Carte sintattiche per le espressioni accettate dal parser.

```
                             ST:='N';
                             READLN
                           END
                     (*$I+*)
                   END;
    PAGE(OUTPUT);
    WRITELN('F = ');
    DISPLAY(EN);
    IF ST='S'
      THEN WRITELN(STAMPANTE,CR,CR,'F = ',EN,CR,CR);
    WRITELN('F''= ');
    DISPLAY(OUT1);
    IF ST='S'
      THEN WRITELN(STAMPANTE,'F''= ',OUT1,CR,CR);
    IF ROOT2<>NIL THEN BEGIN
                         WRITELN('F"= ');
                         DISPLAY(OUT2);
                         IF ST='S'
                           THEN WRITELN(STAMPANTE,'F"= ',OUT2,CR,CR);
                       END
                  ELSE WRITELN('F" NON DISPONIBILE');

    WRITE(CR,CR,CR,CR,'(RET) PER CONTINUARE');
    IF ST='S'
      THEN CLOSE(STAMPANTE,LOCK);
    IF EOLN(KEYBOARD) THEN READ(KEYBOARD,ST);
    READLN(KEYBOARD)
  END;  (* FAIVEDERE *)

BEGIN  (* CONTINUA *)
  REPEAT
    PAGE(OUTPUT);
    GOTOXY(3,5);
    WRITELN('CHE VUOI ?',CR,CR);
    WRITELN('1 - CALCOLO VALORI SINGOLI');
```

```
      WRITELN('2 - FUNZ DERIVATE');
      WRITELN('3 - FINE');
      READ(CH1);
      CASE CH1 OF
        '1' : CALC;
        '2' : FAIVEDERE;
      END  (* DEL CASE *)
    UNTIL CH1='3'
  END;  (* CONTINUA *)


PROCEDURE PRESENTA;
  BEGIN
    PAGE(OUTPUT);
    GOTOXY(15,5);
    WRITELN('DERIVATE :');
    WRITELN(CR,CR,CR);
    WRITELN('LE FUNZIONI ACCETTATE SONO :');
    WRITELN('     SIN,COS,TAN,ATAN,EXP,LOG');
    WRITELN(CR);
    WRITELN('GLI OPERANDI ACCETTATI SONO :');
    WRITELN('     + - * / ^');
    GOTOXY(18,22);
    WRITE('(RET) PER CONTINUARE');
    READLN
  END;


BEGIN  (*  MAIN PROGRAM  *)
  CR:=CHR(13);
  BELL:=CHR(7);  (* INIZIALIZZAZIONE *)
  PRESENTA;
  INIZIA;
  CONTINUA
END.

(****************************************************************************)

                     NOME DEL FILE: CARICA.TEXT




(****************************************************************************)
(*<<<<<<<<<<<                       CARICA                        >>>>>>>>>>>*)
(****************************************************************************)


FUNCTION  CARICA(EN:STRI):PTR;  (* CARICA EN NELL'ALBERO PUNTATO *)
  VAR ABORT:BOOLEAN;


PROCEDURE LE;  (* LEGGE SENZA SKIPPARE I BLANK *)
  BEGIN
    CH:=EN[M];
    M:=M+1
  END;

PROCEDURE LEGGI;  (* LEGGE SKIPPANDO I BLANK *)
 BEGIN
  REPEAT
   CH:=EN[M];
   M:=M+1;
  UNTIL CH<>'
 END;


FUNCTION LEGGINUMERO:REAL;  (* RITORNA IL REALE LETTO DA EN *)
 VAR MULTYPLAY:INTEGER;
     ESP,VAL:REAL;
 BEGIN
  ESP:=1;
  VAL:=0;
  IF CH='-'
   THEN BEGIN
          MULTYPLAY:=-1;
          LE
         END
   ELSE MULTYPLAY:=1;
  IF NOT (CH IN['0'..'9','.']) THEN VAL:=1;
  WHILE CH IN ['0'..'9'] DO
     BEGIN
      VAL:=VAL*10;
      VAL:=VAL+(ORD(CH)-ORD('0'));
      LE
     END;
 IF CH='.'
  THEN BEGIN
        LE;
        WHILE CH IN ['0'..'9'] DO
          BEGIN
           ESP:=ESP/10;
           VAL:=VAL+ESP*(ORD(CH)-ORD('0'));
           LE
          END
       END;
  LEGGINUMERO:=VAL*MULTYPLAY
END;


FUNCTION LEGGIFUNZ:FZ;  (* LEGGE UNA FUNZIONE DA EN *)
 CONST K=5;  (* MAX LUNGHEZZA DI UNA FUNZIONE SEMPLICE *)
 VAR   PAROLA:STRING[K];
       PP,I:INTEGER;

  BEGIN
   PAROLA:='';
   PP:=1;
   ABORT:=TRUE;

   WHILE (CH <> ' ') AND (CH <> '(') AND (PP<=K) DO
     BEGIN
       PAROLA:=CONCAT(PAROLA,'*' (* DUMMY *) );
       PAROLA[PP]:=CH;
       LE;
       PP:=PP+1
     END;
   (* HA CARICATO IL NOME DELLA FUNZIONE IN PAROLA *)

   IF PAROLA='SIN'
     THEN BEGIN LEGGIFUNZ:=SENO;ABORT:=FALSE END;
   IF PAROLA='EXP'
     THEN BEGIN LEGGIFUNZ:=EXPON;ABORT:=FALSE END;
   IF PAROLA='LOG'
     THEN BEGIN LEGGIFUNZ:=LGN;ABORT:=FALSE END;
   IF PAROLA='COS'
     THEN BEGIN LEGGIFUNZ:=COSENO;ABORT:=FALSE END;
   IF PAROLA='TAN'
     THEN BEGIN LEGGIFUNZ:=TAN;ABORT:=FALSE END;
   IF PAROLA='ATAN'
     THEN BEGIN LEGGIFUNZ:=ATANG;ABORT:=FALSE END;
   (* LISTA DELLE FUNZIONI ACCETTATE IN INPUT *)

   IF ABORT THEN ERROR;
   IF CH=' ' THEN LEGGI
  END;


FUNCTION ELEMPTR:PTR;
  (* RICONOSCE IL PROSSIMO ELEMENTO IN EN E LO CARICA IN UN SOTTOALBERO
     DI CUI RITORNA IL PUNTATORE *)
 FORWARD;


FUNCTION FACTORPTR:PTR;
  (* RICONOSCE IL PROSSIMO FACTOR IN EN E LO CARICA IN UN SOTTOALBERO
     DI CUI RITORNA IL PUNTATORE *)
 VAR L,P:PTR;

  BEGIN
   L:=ELEMPTR;
   IF CH='^'
    THEN BEGIN
         NEW(P);
         WITH P^ DO
          BEGIN
           TIPO:=OPER;
           OPERAZIONE:=ELEV;
           SX:=L;
           LEGGI;
           DX:=ELEMPTR
          END
         END
    ELSE P:=L;
   FACTORPTR:=P
  END;


FUNCTION TERMPTR:PTR;
  (* RICONOSCE IL PROSSIMO TERM IN EN E LO CARICA IN UN SOTTOALBERO
     DI CUI RITORNA IL PUNTATORE *)
 VAR L,P:PTR;

  BEGIN
   L:=FACTORPTR;
   WHILE CH IN ['*','/'] DO
     BEGIN
      NEW(P);
      WITH P^ DO
       BEGIN
        TIPO:=OPER;
        IF CH='*'
         THEN OPERAZIONE:=PER
         ELSE OPERAZIONE:=DIVI;
        SX:=L;
        LEGGI;
        DX:=FACTORPTR
       END;
      L:=P
     END;
    TERMPTR:=L
  END;


FUNCTION EXPRPTR:PTR;
  (* RICONOSCE IL PROSSIMO EXPRESSION IN EN E LO CARICA IN UN SOTTOALBERO
     DI CUI RITORNA IL PUNTATORE *)
 VAR L,P:PTR;

  BEGIN
   L:=TERMPTR;
   WHILE CH IN ['+','-'] DO
     BEGIN
      NEW(P);
      WITH P^ DO
       BEGIN
        TIPO:=OPER;
        IF CH='+'
         THEN OPERAZIONE:=PIU
         ELSE OPERAZIONE:=MENO;
        SX:=L;
        LEGGI;
        DX:=TERMPTR
       END;
      L:=P
     END;
    EXPRPTR:=L
  END;


FUNCTION ELEMPTR;
 VAR P:PTR;
     NUMERO:REAL;
     OT,LNU,LF:BOOLEAN;

  BEGIN
   OT:=TRUE;
   LNU:=FALSE;
   LF:=FALSE;
   NEW(P);
   WITH P^ DO
    BEGIN
     IF CH IN ['0'..'9','-']
      THEN BEGIN
            TIPO:=NUM;
            VALORE:=LEGGINUMERO;
            LNU:=TRUE
           END
      ELSE BEGIN
            CASE CH OF
                  'X':BEGIN
                       TIPO:=VR;
                       OT:=FALSE
                      END;
                  '(':BEGIN
                       LEGGI;
                       P:=EXPRPTR;
                       IF CH<>')'
                         THEN ERROR1;
                       OT:=FALSE
                      END;
             END;
            IF OT
             THEN BEGIN
                   TIPO:=FUNZ;
                   FUNZIONE:=LEGGIFUNZIONE;
                   ARG:=ELEMPTR;
                   LF:=TRUE
                  END
           END
    END;
   IF (NOT(LNU OR LF)) OR (CH=' ')
     THEN LEGGI;
   ELEMPTR:=P
  END;


  BEGIN (* CARICA *)
    ABORT:=FALSE;
    OUT:='';
    M:=1;
    LEGGI;
    CARICA:=EXPRPTR;
  END; (* CARICA *)


                     NOME DEL FILE: DERIVA


(****************************************************************************)
(*<<<<<<<<<<<                       SCRIVI                        >>>>>>>>>>>*)
(****************************************************************************)


PROCEDURE SCRIVINUM(X:REAL);  (* SCRIVE X SU OUT *)
  CONST  FIXPT=1000;
  VAR    INTERO:STRING;
```

```
  BEGIN
    IF X<0 THEN
      BEGIN
        OUT:=CONCAT(OUT,'-');
        X:=ABS(X)
      END;
    STR(TRUNC(X),INTERO);
    OUT:=CONCAT(OUT,INTERO);
    IF X<>TRUNC(X)
      THEN  BEGIN
              STR(ABS(ROUND((X-TRUNC(X))*FIXPT)),INTERO);
              OUT:=CONCAT(OUT,'.',INTERO);
            END
  END;

PROCEDURE SCRIVI(INIZIO:PTR);
              (* VISITA L'ALBERO INORDER E LO SCRIVE SU OUT *)
  BEGIN
   CASE INIZIO^.TIPO OF
        VR:OUT:=CONCAT(OUT,'X');
        NUM:SCRIVINUM(INIZIO^.VALORE);
        OPER:BEGIN
               IF INIZIO^.OPERAZIONE IN [PIU,MENO,ELEV]
                THEN BEGIN
                       OUT:=CONCAT(OUT,'(');
                       SCRIVI(INIZIO^.SX);
                       CASE INIZIO^.OPERAZIONE OF
                             PIU:OUT:=CONCAT(QUT,'+');
                             MENO:OUT:=CONCAT(OUT,'-');
                             ELEV:OUT:=CONCAT(OUT,')^(')
                            END;
                       SCRIVI(INIZIO^.DX);
                       OUT:=CONCAT(OUT,')');
                      END
                ELSE BEGIN
                       SCRIVI(INIZIO^.SX);
                       CASE INIZIO^.OPERAZIONE OF
                            PER:BEGIN
                                  OUT:=CONCAT(OUT,'*');
                                  SCRIVI(INIZIO^.DX)
                                 END;
                            DIVI:BEGIN
                                   OUT:=CONCAT(OUT,'/(');
                                   SCRIVI(INIZIO^.DX);
                                   OUT:=CONCAT(OUT,')');
                                  END
                         END
                      END
             END;
        FUNZ:BEGIN
              CASE INIZIO^.FUNZIONE OF
                      SENO   :OUT:=CONCAT(OUT,'SIN');
                      COSENO:OUT:=CONCAT(OUT,'COS');
                      TAN    :OUT:=CONCAT(OUT,'TAN');
                      LGN    :OUT:=CONCAT(OUT,'LOG');
                      EXPON  :OUT:=CONCAT(OUT,'EXP');
                      ATANG  :OUT:=CONCAT(OUT,'ATAN')
                      (* E, VOLENDO, ALTRE *)
               END;
              OUT:=CONCAT(OUT,'(');
              SCRIVI(INIZIO^.ARG);
              OUT:=CONCAT(OUT,')');
             END;
      END
  END;

(**************************************************************************)
(*<<<<<<<<<<                         DERIVA                     >>>>>>>>>>*)
(**************************************************************************)

PROCEDURE DERIVA(RADICE:PTR);

  (* RICAVA RICORSIVAMENTE LA DERIVATA, SCRIVENDOLA SU OUT, VISITANDO
     L' ALBERO CHE INIZIA IN RADICE *)

 PROCEDURE DERELEV; (* TRATTA LA DERIVATA DI UNA ELEVAZIONE,
                       CON VARI SOTTOCASI *)

  PROCEDURE DEREOP;
   (* USATA SE A SINISTRA & A DESTRA DI ^ VI SONO OPERAZIONIE/O FUNZIONI *)

   BEGIN
    WITH RADICE^ DO
     BEGIN
      OUT:=CONCAT(OUT,'(');
      SCRIVI(SX);
      OUT:=CONCAT(OUT,')^(');
      SCRIVI(DX);
      OUT:=CONCAT(OUT,')*((');
      SCRIVI(DX);
      OUT:=CONCAT(OUT,')*(');
      DERIVA(SX);
      OUT:=CONCAT(OUT,')/(');
      SCRIVI(SX);
      OUT:=CONCAT(OUT,')+(');
      DERIVA(DX);
      OUT:=CONCAT(OUT,')*LOG(');
      SCRIVI(SX);
      OUT:=CONCAT(OUT,'))')
     END (* WITH *)
   END;

  PROCEDURE RIVA;
   (* CONSIDERA I CASI IN CUI A SINISTRA DI ^ VI SONO OPERAZIONI O FUNZIONI *)

   BEGIN
    WITH RADICE^ DO
     BEGIN
      CASE DX^.TIPO OF
       NUM:BEGIN
             SCRIVINUM(DX^.VALORE);
             OUT:=CONCAT(OUT,'*(');
             DERIVA(SX);
             OUT:=CONCAT(OUT,')*(');
             SCRIVI(SX);
             OUT:=CONCAT(OUT,')^');
             SCRIVINUM(DX^.VALORE-1)
            FND;
       VR:BEGIN
            OUT:=CONCAT(OUT,'(');
            SCRIVI(SX);
            OUT:=CONCAT(OUT,')^X*(X/(');
            SCRIVI(SX);
            OUT:=CONCAT(OUT,')+LOG(');
            SCRIVI(SX);
            OUT:=CONCAT(OUT,'))');
          END;
      OPER,FUNZ:DEREOP
     END (* CASE ]I[ *)
    END (* WITH *)
  END;

  BEGIN (* DERELEV *)

    WITH RADICE^ DO
     BEGIN
```

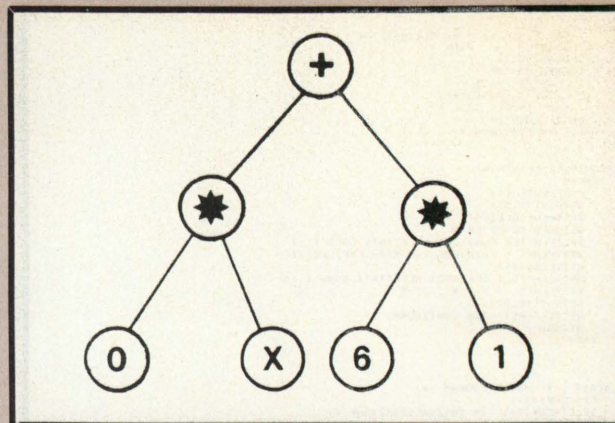



Figura 2 - Esempio di Albero rappresentante l'espressione 0 ★ X + 6 ★ 1

```
      CASE SX^.TIPO OF
           NUM:CASE DX^.TIPO OF
                     NUM:OUT:=CONCAT(OUT,'0');
                     VR:BEGIN
                          SCRIVINUM(SX^.VALORE);
                          OUT:=CONCAT(OUT,'^X*LOG ');
                          SCRIVINUM(SX^.VALORE)
                         END;
                     FUNZ,OPER:BEGIN
                                 SCRIVINUM(SX^.VALORE);
                                 OUT:=CONCAT(OUT,'^(');
                                 SCRIVI(DX);
                                 OUT:=CONCAT(OUT,')*LOG ');
                                 SCRIVINUM(SX^.VALORE);
                                 OUT:=CONCAT(OUT,'*(');
                                 DERIVA(DX);
                                 OUT:=CONCAT(OUT,')')
                                END
                    END;
           VR:CASE DX^.TIPO OF
                    NUM:IF DX^.VALORE=1
                          THEN OUT:=CONCAT(OUT,'1')
                          ELSE IF DX^.VALORE=0
                                 THEN OUT:=CONCAT(OUT,'0')
                                 ELSE
                                BEGIN
                                 SCRIVINUM(DX^.VALORE);
                                 OUT:=CONCAT(OUT,'*X^');
                                 SCRIVINUM(DX^.VALORE-1)
                                END;
                    VR:OUT:=CONCAT(OUT,'X^X*(1+LOG X)');
                    FUNZ,OPER:BEGIN
                                OUT:=CONCAT(OUT,'X^(');
                                SCRIVI(DX);
                                OUT:=CONCAT(OUT,')*(');
                                SCRIVI(DX);
                                OUT:=CONCAT(OUT,'/X+LOG X*(');
                                DERIVA(DX);
                                OUT:=CONCAT(OUT,'))')
                               END
                   END;
              OPER,FUNZ:RIVA
            END (* CASE ]I[ *)
     END;(* WITH *)
  END; (* DERELEV *)

  PROCEDURE TOOLONG;
    (* ELENCO DELLE DERIVATE DI FUNZIONI ELEMENTARI *)

    BEGIN
      WITH RADICE^ DO
        BEGIN
          OUT:=CONCAT(OUT,'(');
          DERIVA(ARG);
          OUT:=CONCAT(OUT,')*');
          CASE FUNZIONE OF
                SENO:OUT:=CONCAT(OUT,'COS');
                COSENO:OUT:=CONCAT(OUT,'-1*SIN');
                LGN:OUT:=CONCAT(OUT,'1/');
                TAN:OUT:=CONCAT(OUT,'1/(COS');
                ATANG:OUT:=CONCAT(OUT,'1/(1+');
                EXPON:OUT:=CONCAT(OUT,'EXP')
                (* ECCETERA *)
           END;
          OUT:=CONCAT(OUT,'(');
          SCRIVI(ARG);
          OUT:=CONCAT(OUT,')');
          IF FUNZIONE=TAN THEN OUT:=CONCAT(OUT,')^2');
          IF FUNZIONE=ATANG THEN OUT:=CONCAT(OUT,'^2)');
        END
    END;

  BEGIN (* DI DERIVA *)
   WITH RADICE^ DO
    BEGIN
     CASE TIPO OF  (* CONSIDERA I VARI TIPI DI NODO *)
      VR: OUT:=CONCAT(OUT,'1');
      NUM: OUT:=CONCAT(OUT,'0');
      FUNZ:TOOLONG;
      OPER:CASE OPERAZIONE OF  (* CONSIDERA I VARI CASI.DI OPERATORI *)
                PIU:BEGIN
                      DERIVA(SX);
                      OUT:=CONCAT(OUT,'+');
                      DERIVA(DX)
                     END;
                MENO:BEGIN
                       DERIVA(SX);
                       OUT:=CONCAT(OUT,'-');
                       DERIVA(DX)
                      END;
                PER:BEGIN
                      OUT:=CONCAT(OUT,'(');
                      DERIVA(SX);
                      OUT:=CONCAT(OUT,')*(');
                      SCRIVI(DX);
                      OUT:=CONCAT(OUT,')+(');
                      SCRIVI(SX);
                      OUT:=CONCAT(OUT,')*(');
                      DERIVA(DX);
                      OUT:=CONCAT(OUT,')');
                     END;
                DIVI:BEGIN
                      OUT:=CONCAT(OUT,'((');
                      DERIVA(SX);
                      OUT:=CONCAT(OUT,')*(');
                      SCRIVI(DX);
                      OUT:=CONCAT(OUT,')-(');
                      SCRIVI(SX);
                      OUT:=CONCAT(OUT,')*(');
```

```
                    DERIVA(DX);
                    OUT:=CONCAT(OUT,'))/((');
                    SCRIVI(DX);
                    OUT:=CONCAT(OUT,')^2)');
                   END;
                ELEV:DERELEV

           END (* CASE 1 *)
       END (* CASE 0 *)
      END (* WITH *)
    END;


                  NOME DEL FILE : CALCOLO

(****************************************************************************)
(*<<<<<<<<<<                       CALCOLO                      >>>>>>>>>>*)
(****************************************************************************)


FUNCTION CALCOLO(INIZIO:PTR;VALORIX:REAL):REAL;
  VAR  PROVV,CD,CS:REAL;
       IKE:INTEGER;


  PROCEDURE GESTERR(TIPO:TIPOERRORE);
    BEGIN
      IF ERRORE=NESSUNO
        THEN ERRORE:=TIPO;
      CALCOLO:=0;
      EXIT(CALCOLO)
    END;

  FUNCTION TANG(X:REAL):REAL;
    VAR P:REAL;
    BEGIN
      P:=COS(X);
      IF ABS(P)<UNDER THEN GESTERR(OVERFLOW)
                     ELSE TANG:=SIN(X)/P
    END;


BEGIN (* CALCOLO *)

 WITH INIZIO^ DO
   BEGIN
     CASE TIPO OF
       VR:PROVV:=VALORIX;
       NUM:PROVV:=VALORE;

       OPER:CASE OPERAZIONE OF
              PIU:PROVV:=CALCOLO(SX,VALORIX)+CALCOLO(DX,VALORIX);
              MENO:PROVV:=CALCOLO(SX,VALORIX)-CALCOLO(DX,VALORIX);
              PER:PROVV:=CALCOLO(SX,VALORIX)*CALCOLO(DX,VALORIX);
              (* IN QUESTI TRE CASI NON C'E' RISCHIO DI OVERFLOW ! *)
              DIVI:BEGIN
                     CD:=CALCOLO(DX,VALORIX);
                     CS:=CALCOLO(SX,VALORIX);
                     IF CD=0
                       THEN GESTERR(DIVBYZER)
                       ELSE IF ABS(CS)<ABS(CD)  (* TUTTO OK *)
                              THEN PROVV:=CS/CD
                              ELSE IF ABS(CD/CS)<UNDER
                                   (* TROPPO ALTO IL RECIPROCO *)
                                     THEN GESTERR(OVERFLOW)
                                     ELSE PROVV:=CS/CD
                   END;
              ELEV:BEGIN
                     CS:=CALCOLO(SX,VALORIX);
                     CD:=CALCOLO(DX,VALORIX);
                     IF CS<0
                       THEN IF CD<>TRUNC(CD)
                               THEN GESTERR(EXPNEG)
                               ELSE BEGIN
                                      PROVV:=LN(ABS(CS))*CD;
                                      IF PROVV>6*LN(10)
                                        THEN GESTERR(OVERFLOW)
                                        ELSE PROVV:=EXP(PROVV);
                                      IF ODD(TRUNC(CD)) THEN PROVV:=-PROVV;
                                      CALCOLO:=PROVV;
                                      EXIT(CALCOLO)
                                     END
                       ELSE IF CS=0
                              THEN IF CD>0
                                     THEN PROVV:=0
                                     ELSE IF CD=0
                                            THEN PROVV:=1
                                            ELSE GESTERR(DIVBYZER)
                              ELSE BEGIN
                                      PROVV:=LN(CS)*CD;
                                      IF PROVV>6*LN(10)
                                        THEN GESTERR(OVERFLOW)
                                        ELSE PROVV:=EXP(PROVV)
                                    END
                   END
           END;

      FUNZ:CASE FUNZIONE OF
             SENO  :PROVV:=SIN(CALCOLO(ARG,VALORIX));
             COSENO:PROVV:=COS(CALCOLO(ARG,VALORIX));
             LGN   :BEGIN
                      PROVV:=CALCOLO(ARG,VALORIX);
                      IF PROVV<=0 THEN GESTERR(LOGNEG)
                                  ELSE PROVV:=LN(PROVV)
                    END;
             TAN   :PROVV:=TANG(CALCOLO(ARG,VALORIX));
             ATANG :PROVV:=ATAN(CALCOLO(ARG,VALORIX));
             EXPON :PROVV:=EXP(CALCOLO(ARG,VALORIX));
             (* ED ALTRE, VOLENDO *)
           END
     END (* CASE *)
   END; (* WITH *)
 CALCOLO:=PROVV
END; (* CALCOLO *)


                  NOME DEL FILE: SEMPLIFICA

 (**********************************************************************)
 (*<<<<<<<<                    SEMPLIFICA                    >>>>>>>>*)
 (**********************************************************************)


 PROCEDURE SEMPLIFICA(VAR QUI:PTR);

 (*  HA IL COMPITO DI SEMPLIFICARE UN PO' LA FUNZIONE *)


   VAR BUFF:PTR;  (* BUFFERIZZA IL PUNTATORE QUI CORRETTO *)


   PROCEDURE ERR(S:STRING);  (* GESTISCE GLI ERRORI CERTI *)
     BEGIN
       WRITELN(BELL,CR,'ATTENTO: LA FUNZIONE CHE STAI DERIVANDO',CR,
               'NON E'' DEFINITA SU TUTTO L''ASSE REALE',CR,
               'A CAUSA DI');
       WRITELN(S);
       EXIT(PROGRAM)
     END;

 PROCEDURE DAFARE;  (* SEMPLIFICA LA PARTE DELL'ESPRESSIONE CHE
                       CONTIENE L'ELEVAZIONE *)
   VAR A:REAL;
   BEGIN
     WITH BUFF^ DO
       BEGIN
         IF (SX^.TIPO=NUM) AND (DX^.TIPO=NUM)
         (* CIOE' SE ENTRAMBI GLI OPERANDI SONO NUMERI *)
           THEN BEGIN (* CALCOLALI,SE PUOI,ED ESCI *)
                  ERRORE:=NESSUNO;
                  A:=CALCOLO(QUI,1 (*DUMMY*) );
                  CASE ERRORE OF
                    DIVBYZER : ERR('DIVISIONE PER ZERO');
                    EXPNEG   : ERR('BASE NGATIVA');
                    OVERFLOW : EXIT(DAFARE)
                  END;  (* CASE *)
                  QUI:=SX;
                  SX^.VALORE:=A;
                  EXIT(DAFARE)
                END;
         IF SX^.TIPO=NUM  (* NUMERO SOLO A SINISTRA *)
           THEN BEGIN IF (SX^.VALORE=1) OR (SX^.VALORE=0)
                        THEN QUI:=SX (* TUTTO UGUALE A 1 O 0  *)
                END
           ELSE IF DX^.TIPO=NUM  (* NUMERO SOLO A DESTRA *)
                  THEN IF DX^.VALORE=0  (* TUTTO UGUALE AD 1 *)
                         THEN BEGIN
                                DX^.VALORE:=1;
                                QUI:=DX
                              END
                         ELSE IF DX^.VALORE=1
                                THEN QUI:=SX (* BYPASSALO *)
       END  (* WITH *)
   END;  (* DAFARE *)


 PROCEDURE MOLTIP;

   BEGIN
     WITH BUFF^ DO
       BEGIN
         IF (SX^.TIPO=NUM) AND (DX^.TIPO=NUM)
         (* CIOE' SE ENTRAMBI GLI OPERANDI SONO NUMERI *)
           THEN BEGIN (* MOLTIPLICALI ED ESCI *)
                  SX^.VALORE:=SX^.VALORE*DX^.VALORE;
                  QUI:=SX;
                  EXIT(MOLTIP)
                END;
         IF SX^.TIPO=NUM  (* NUMERO SOLO A SINISTRA *)
           THEN BEGIN IF SX^.VALORE=1
                        THEN QUI:=DX (* BYPASSALO *)
                        ELSE IF SX^.VALORE=0
                               THEN QUI:=SX (* TUTTO NULLO *)
                END
           ELSE IF DX^.TIPO=NUM  (* NUMERO SOLO A DESTRA *)
                  THEN IF DX^.VALORE=1
                         THEN QUI:=SX (* BYPASSALO *)
                         ELSE IF DX^.VALORE=0
                                THEN QUI:=DX (* TUTTO NULLO *)
       END  (* WITH *)
   END;  (* MOLTIP *)


 PROCEDURE PIUMENO;

   BEGIN
     WITH BUFF^ DO
       BEGIN
         IF (SX^.TIPO=NUM) AND (DX^.TIPO=NUM)
         (* CIOE' SE ENTRAMBI GLI OPERANDI SONO NUMERI *)
           THEN BEGIN (* ADDIZIONALI ED ESCI *)
                  CASE OPERAZIONE OF
                    PIU :SX^.VALORE:=SX^.VALORE+DX^.VALORE;
                    MENO:SX^.VALORE:=SX^.VALORE-DX^.VALORE
                  END; (* DEL CASE *)
                  QUI:=SX;
                  EXIT(PIUMENO)
                END;
         IF SX^.TIPO=NUM  (* NUMERO SOLO A SINISTRA *)
           THEN BEGIN
                  IF SX^.VALORE=0
                    THEN BEGIN
                           IF OPERAZIONE=PIU
                             THEN QUI:=DX (* BYPASSALO *)
                             ELSE (* C'E' UN MENO *)
                               BEGIN
                                 SX^.VALORE:=-1;
                                 QUI^.OPERAZIONE:=PER  (* METTE UN PER -1  *)
                               END
                         END
                END
           ELSE IF DX^.TIPO=NUM  (* NUMERO SOLO A DESTRA *)
                  THEN IF DX^.VALORE=0
                         THEN QUI:=SX (* BYPASSALO *)
       END  (* WITH *)
   END;  (* PIUMENO *)


 PROCEDURE FRATTO;

   BEGIN
     WITH BUFF^ DO
       BEGIN
         IF DX^.TIPO=NUM THEN IF DX^.VALORE=0
                                THEN ERR('DIVISIONE PER ZERO');
           (* FUNZIONE NON DEFINITA SU TUTTO L'ASSE REALE *)
         IF (SX^.TIPO=NUM) AND (DX^.TIPO=NUM)
         (* CIOE' SE ENTRAMBI GLI OPERANDI SONO NUMERI *)
           THEN BEGIN (* DIVIDILI E ESCI *)
                  SX^.VALORE:=SX^.VALORE/DX^.VALORE;
                  QUI:=SX;
                  EXIT(FRATTO)
                END;
         IF SX^.TIPO=NUM  (* NUMERO SOLO A SINISTRA *)
           THEN BEGIN IF SX^.VALORE=0
                        THEN QUI:=SX (* TUTTO NULLO *)
                END
           ELSE IF DX^.TIPO=NUM  (* NUMERO SOLO A DESTRA *)
                  THEN IF DX^.VALORE=1
                         THEN QUI:=SX (* BYPASSALO *)
       END  (* WITH *)
   END;  (* FRATTO *)


BEGIN  (* DI SEMPLIFICA *)
  BUFF:=QUI;  (* BUFFERIZZA QUI PER IL WITH STATEMENT
                 (SE NO NON POTREI ASSEGNARE UN VALORE A QUI) *)
  WITH BUFF^ DO
    CASE TIPO OF

      OPER : BEGIN
               SEMPLIFICA(SX);
               SEMPLIFICA(DX);
               (* SEMPLIFICA PRIMA I DUE OPERANDI *)
               CASE OPERAZIONE OF
                 PER      :MOLTIP;
                 PIU,MENO:PIUMENO;
                 DIVI     :FRATTO;
                 ELEV     :DAFARE
               END;  (* CASE INTERNO *)
             END;

      FUNZ : BEGIN
               SEMPLIFICA(ARG);
               IF ARG^.TIPO=NUM
                 THEN BEGIN
                        ARG^.VALORE:=CALCOLO(QUI,VALORE);
                        QUI:=ARG
                      END
             END
    END  (* CASE ESTERNO *)
END;  (*  DI SEMPLIFICA  *)
```

Usare il sistema operativo CP/M

IL LIBRO
Il sistema operativo CP/M è stato progettato per rendere semplice l'uso di un microcomputer. Questo libro vi renderà semplice l'uso del CP/M. (Le versioni esaminate del CP/M sono il CP/M 1.4-il CP/M 2.2. e il nuovo sistema operativo multiutente MP/M) La maggior parte di utenti di microcomputer dovrà, infatti, un giorno o l'altro, fare ricorso al CP/M, disponibile su quasi tutti i computer basati sui microprocessori 8080 e Z80, come pure su certi sistemi utilizzanti il 6502. Il libro, senza presupporre alcuna conoscenza di un calcolatore, inizia con la descrizione, passo-passo delle procedure di inizializzazione del sistema: accensione, inserimento dei dischetti, esecuzione delle più comuni operazioni su file, compresa la duplicazione dei dischetti. Prosegue con il PIP (programma di trasferimento dei file), il DDT (programma di messa a punto) e ED (programma editor). Per entrare sempre più, fornendo numerosi consigli pratici, all'interno del CP/M e delle sue operazioni, al fine di comprenderne appieno le risorse ed eventualmente dare gli strumenti per successive modifiche.

SOMMARIO
Introduzione al CP/M e all'MP/M-Le caratteristiche del CP/M e dell'MP/M-Gestione dei file con PIP-L'uso dell'editor-Dentro al CP/M e all'MP/M-Guida di riferimento ai comandi e ai programmi del CP/M e dell'MP/M-Consigli pratici-Il futuro-messaggi comuni di errore-tabella di controllo di ED-nomi dei dispositivi di PIP-riassunti dei comandi-parole chiave di PIP-parametri di PIP-tasti di controllo per la digitazione dei comandi-tipi di estensione-lista dei materiali-organizzazione della stanza del calcolatore-verifiche in caso di errore-regole di base per la localizzazione dei guasti.

Pagg. 320 Cod. 510P
L.22.000 (Abb. L.19.800)

Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista.

GRUPPO EDITORIALE JACKSON
Divisione Libri

Amper Interpreter:

ovvero quando il BASIC non basta più

di A. Stecchina

Su mele e ampere galoppa l'alfiere

Il grazioso simbolo & che quasi nessuno sa scrivere e che molti chiamano "e commerciale", si chiama internazionalmente ampersand. Esso è una delle caratteristiche più interessanti del BASIC Applesoft, in quanto ne permette l'estensione dei comandi e la creazione "ex novo" di altri per applicazioni particolari, ad esempio grafica, editing, applicazioni gestionali, ecc.
L'uso dell' ampersand non viene in alcun modo esemplificato nel manuale Applesoft, probabilmente per questioni di spazio, e questo è un peccato in quanto non permette ai potenziali utenti di avvalersi di questo formidabile comando.
Fino ad ora in Italia non sono apparsi articoli che utilizzano l'ampersand , ad eccezione dell' ottimo articolo di Cerofolini apparso su **Bit** n.31 *(Program Editor)*.
In questo numero creeremo un interprete per comandi ampersand in grado di riconoscere ben 127 istruzioni diverse e nei prossimi articoli realizzeremo numerosi comandi per l'animazione, l'impaginazione, ecc.
Prepariamoci dunque a questa lunga cavalcata che ci porterà nei meandri del BASIC e del DOS per culminare nella creazione del Super Apple (il mitico).

Leggimi, non te ne pentirai

Poichè questo è il primo programma in linguaggio macchina che scrivo su **Bit**, vorrei fare una premessa.
Quelli di voi che hanno già buttato lo sguardo vorace sul listing del programma si saranno accorti che è scritto in Assembler e molti avranno già pensato: "che peccato"; preparandosi a passare al prossimo articolo.
Questo è purtroppo un atteggiamento abbastanza comune tra i lettori che non conoscono il linguaggio macchina e tra coloro che sono alle prime armi. Questo atteggiamento di diffidenza e di timore nei riguardi del linguaggio macchina lo abbiamo avuto un po' tutti quando abbiamo cominciato. Non c'è quindi nulla di cui vergognarsi, bisogna però cercare di percepire dagli articoli relativi al linguaggio macchina il più possibile e ripromettersi di rileggerli quando si saranno acquisite altre nozioni. Una delle differenze che contraddistinguono questo articolo dalla maggior parte di quelli dedicati al linguaggio macchina apparsi finora è che...
Questo articolo è scritto anche per coloro che non sanno assolutamente nulla di linguaggio macchina.
Tutti i lettori digiuni di linguaggio macchina che hanno avuto la pazienza di leggermi fin qui, troveranno di che soddisfare un po' della loro sete di sapere.
Non posso ovviamente fare un corso di Assembler in queste poche pagine, ma vorrei dare un "feeling" di quello che è un programma scritto in Assembler.
Vi saranno purtroppo alcune parti non comprensibili a tutti, ma buona parte dell'articolo è stata scritta espressamente per coloro che non conoscono nulla dell' Assembler.

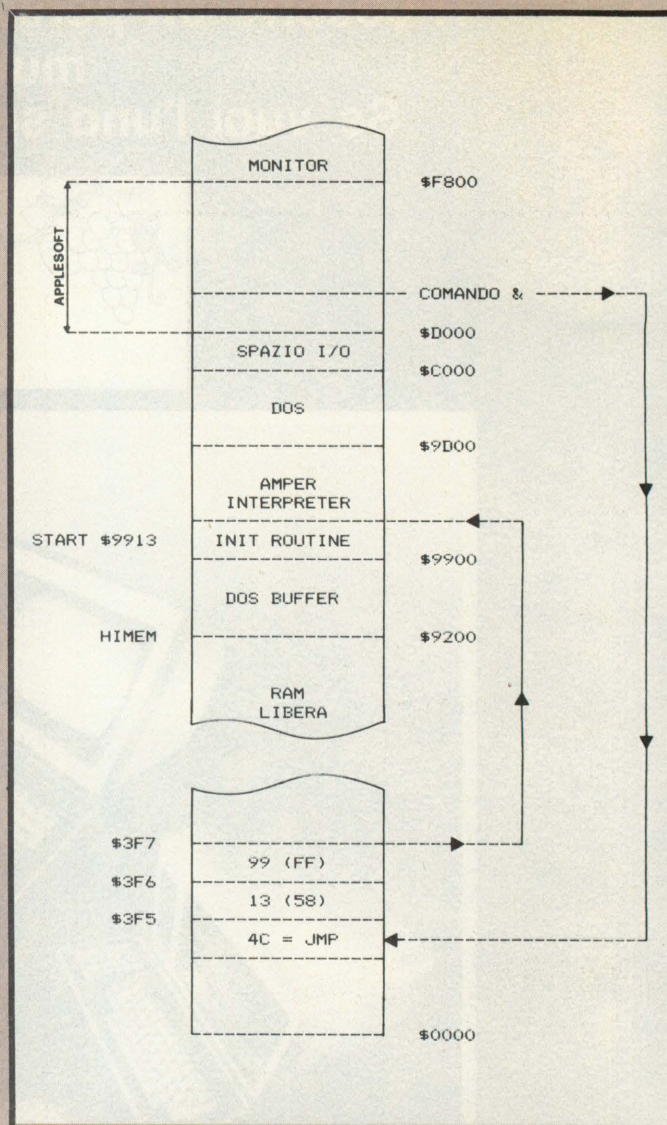



Figura 1 - Ecco cosa avviene quando è risultato il comando in presenza dell'Amper-Interpreter. Il BASIC salta alla locazione $ 3F5 dove incontra un JMP alla routine di gestione del comando (default $FF58). Nel nostro caso l'indirizzo è $9913 (Amper-Interpreter).

Assembler per tutti

Il programma è stato scritto usando l'SC macro Assembler un potente Assembler che (come dice il nome) permette l'uso delle macro. Vedremo dopo cosa sono le macro, per ora vediamo cos'è un Assembler.
Le istruzioni del linguaggio macchina sono dei numeri esadecimali da un byte.
Ovviamente nessuno si ricorda a quale numero corrisponde una determinata operazione e pertanto sono stati creati dei programmi che accettano in input la sigla del comando del linguaggio macchina e la tra« sformano nel numero corrispondente.
Ad esempio per caricare l'accumulatore con *un certo* numero esadecimale, si deve usare il codice *$A9*, ma se si ha un Assembler basta dare al computer il comando *LDA* che che è un'abbreviazione di load Accumulator.
Queste abbreviazioni hanno un'utilità mnemonica e per-

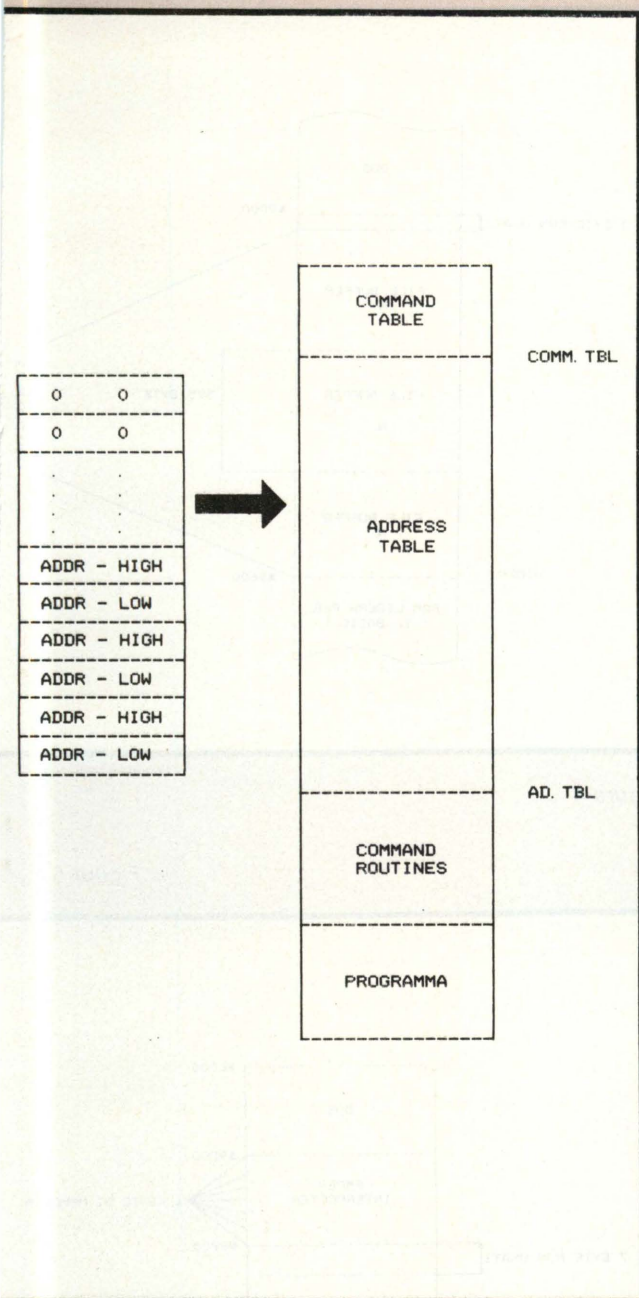
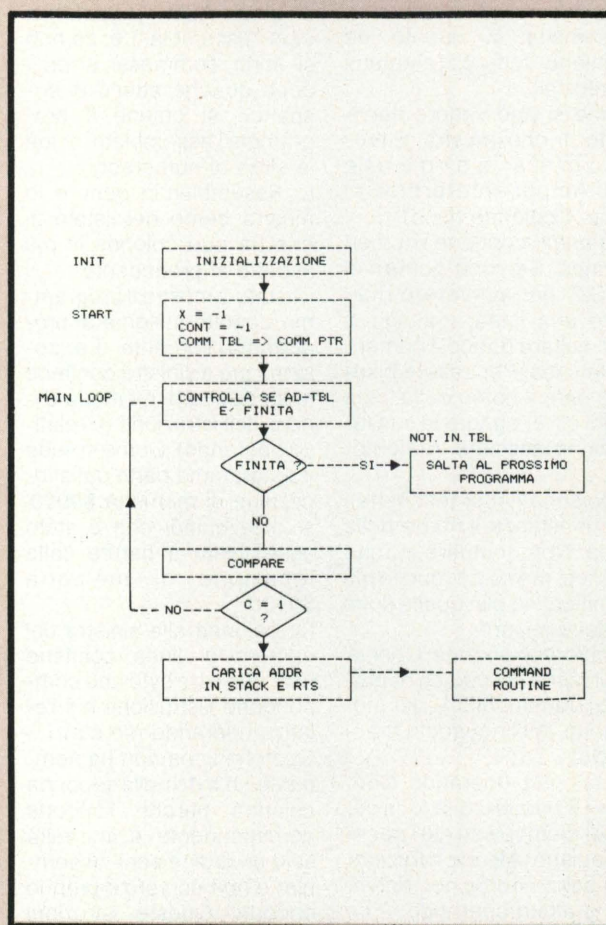



Figura 2

Figura 3

tanto vengono chiamati codici mnemonici.
Tradurre i codici mnemonici nei numeri loro corrispondenti è dunque lo scopo primario di un Assembler.
Programmare in linguaggio macchina o programmare in Assembler vengono considerati quasi sempre dei sinonimi poichè qualunque programma in linguaggio assoluto appena un pochino complesso viene scritto senz'altro in Assembler.
Il programma scritto in Assembler contenente i codici mnemonici si chiama programma sorgente, il programma in linguaggio macchina vero e proprio (che è in sostanza una sfilza di numeracci in esadecimale) si chiama programma oggetto e viene generato automaticamente dall' Assembler a partire dal programma sorgente. Questo processo si chiama "Assemblare". Uno scopo secondario (ma importantissimo e irrinunciabile) dell' Assembler è anche quello della sostituzione delle variabili e delle costanti tramite etichette (Label), la possibilità cioè di dare un nome a delle locazioni di memoria o a dei numeri. Ad esempio l'inventore di Invaders, quando ha scritto il celeberrimo programma in Assembler avrà avuto certamente una locazione di memoria dove era memorizzato un punteggio del giocatore (l'agognato *SCORE* che tutti cercano sempre di migliorare). Avrà quindi detto all' Assembler che la variabile *SCORE* era uguale a *$ABCD* (tanto per fare un esempio), e poi se ne sarà completamente dimenticato per tutto il resto del programma , chiamando in seguito quella locazione sempre col nome *SCORE*.

Chiarita per sommi capi la natura di un Assembler , vediamo ora cos'è un macro Assembler.

Micro macro

Un macro Assembler permette l'uso di macro, che in parole povere sono spezzoni di programma con delle parti lasciate in bianco.
Quando una macro viene chiamata gli vengono passati di solito dei parametri (variabili, indirizzi, costanti, ecc) che prendono il posto delle parti lasciate in bianco e lo spezzone del programma così ottenuto viene inserito nel programma principale.
Mostrerò più avanti dove sono le macro nell' Amper-Interpreter.
L'Amper-Interpreter è un ti-

pico esempio di programma Assembler in quanto ne contiene tutti gli elementi essenziali.
Come si può vedere dal listato 1 che mostra il programma sorgente dell'Amper-Interpreter ci sono 4 colonne (field).
I colonna: contiene i numeri di linea. Servono come nel BASIC per scrivere e rivedere una linea, ma non si può saltare dando il numero della linea. Per saltare bisogna dare il nome della linea (vedi oltre) oppure la sua locazione assoluta in memoria.
II colonna: label (etichetta). È in sostanza il nome della linea. Non compare in tutte le linee, ma solo in quelle più significative o in quelle dove si deve saltare.
III colonna: opcode. Contiene il nome del codice operativo (mnemonico) del comando del linguaggio macchina.
IV colonna: operando. Contiene il numero o la locazione di memoria su cui l'opcode agisce. Alcune istruzioni, ma sono poche, non richiedono alcun operando.
Un esempio si può vedere nella linea *1450* dove compare l'istruzione *ASL*.
V colonna: commento. Contiene un eventuale commento su ciò che viene fatto in quella linea. L'uso di questa colonna (come i *REM* in BASIC del resto) è affidato al buon gusto e alle capacità descrittive del programmatore.
Se si voul fare una linea fatta solo di commenti, bisogna farla precedere da un ★.
Si può notare che molti nomi di variabili contengono dei punti all'interno (es. AD.TBL). Questo è un semplice trucco per ottenere nomi di variabili formati da più parole usando il punto al posto dello spazio.
Il programma nella forma del listato 1 si chiama, ripetiamo, il programma sorgente. Se si è soddisfatti del programma sorgente si ordina all' Assembler di assemblarlo.
Questo in genere si fa con un comando diretto che si chiama ASM (ASseMble, cioè "assembla") e, se non si sono commessi errori , dopo quache attimo di suspance si ottiene il programma assemblato (cioè la sfilza di numeracci).
L' Assembler in genere lo mostra come nel listato 2, che ha due colonne in più rispetto al precedente.
Il listato 2 mostra il programma oggetto insieme al programma sorgente. La colonna più a sinistra contiene la locazione di memoria d'inizio dell'istruzione (e relativo operando). Come si vede il programma parte dalla locazione di memoria *$9900*, si dice quindi che è stato assemblato a partire dalla locazione di memoria *$9900*.
La colonna alla sinistra del numero di linea contiene uno, due o tre byte che compongono l'istruzione e il relativo operando (se c'è).
Qualche linea non ha nemmeno un byte nella seconda colonna perchè l'opcode corrispondente è in realtà solo un ordine per l' Assembler e non un vero e proprio opcode. Queste istruzioni per l' Assembler si chiamano perciò *pseudo opcode*. In questo caso dunque non viene generato alcun programma oggetto dall' Assembler , ma solo eseguito un ordine particolare. Ad esempio l' opcode *.EQ* serve a dare un nome ad una variabile o ad una costante o ad una locazione di memoria che poi verrà richiamata in seguito. Un esempio di ciò si ha nelle linee *1240* e *1430*, dove alla variabile *COUNT* viene assegnata la locazione di memoria *$F9* (linea *1240*) e poi questa variabile viene incrementata (linea *1430*).
È questo comando che ci permette di dare dei nomi a delle variabili o a delle costanti come accennato in precedenza.
Si noti la seconda colonna della linea del listato 2, che contiene il byte che rappresenta l'istruzione *INC* e il byte che rappresenta la locazione di memoria della variabile CONT. Faccio no-

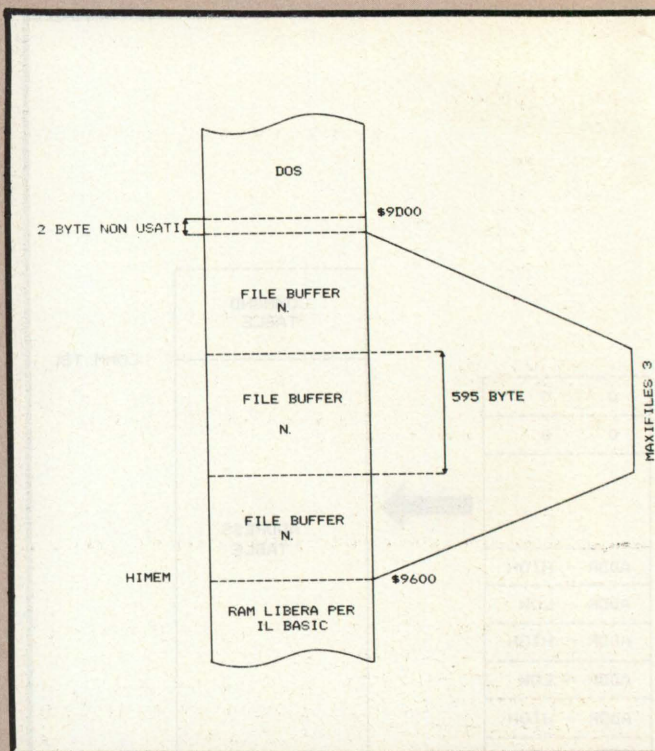



Figura 4

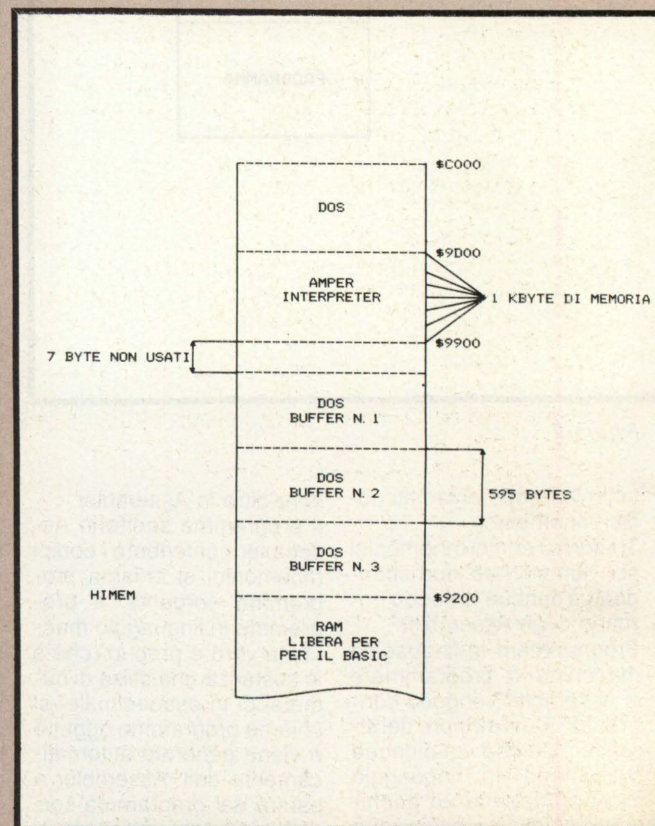
Figura 5

```
CALL -151                          per passare al monitor

9D01 : 98
A7D4 G
9D0D : 99
9E77 : 13 99
9DD3 : B3 B6
B7E0 : 1F
B71A : 08
B763 : 08

CTRL - C                           per tornare al BASIC

BLOAD DOS1. CODE
BLOAD DOS2. CODE
BLOAD AMP. CODE

scrivere (o caricare) un programma "HELLO" e battere:
INIT HELLO (oppure un altro nome)
```

Figura 6

tare che le pseudo istruzioni dell' S-C macro Assembler sono precedute da un puntino, ma questa non è una regola generale negli Assembler, ciò che invece è assolutamente standard è il suddetto significato del comando *EQ* (*EQU* in altri Assembler). Notate anche che l'Assembler fornisce una tabella dei valori corrispondenti alle etichette. Questo è utile per chi legge il programma poichè può trovare subito la locazione di un etichetta particolare senza dover scorrere tutto il listato.
Un'altra pseudo molto importante è la *.OR* (*ORG* in altri Assembler) che dice all' Assembler da quale locazione cominciare ad assemblare il programma.
Ad esempio nella linea *1200* troviamo un *.OR $9900* che indica all' Assembler che noi vogliamo che il programma oggetto sia assemblato a partire dalla locazione di memoria *$9900*.
Nell' Amper-Interpreter vengono usate 2 macro.
Una si chiama *INCD* e l'altra *PNTR*.
Esse vengono chiamate mettendo il segno > (davanti al loro nome e la lista dei parametri da sostituire alle "parti lasciate in bianco" di seguito al nome, separata da uno spazio.
La definizione di queste macro è fatta alle linee *1000-1080* e *1090-1180* del listato 1.
Ogni macro è racchiusa tra le pseudo *.MA* e *.EM* mentre le "parti lasciate in bianco" sono contrassegnate da }n dove n è un numero.
Un altra differenza che potete notare tra i due listati è che in corrispondenza ai comandi *>INCD*... e *>PNTR*... del listato 1 (linee *1700* e *1350* ad esempio) ci sono nel listato 2 alcune linee di programma che costituiscono le macro suddette.
Nel listato 2 le macro sono state messe nel programma laddove vengono chiamate e "le parti lasciate in bianco" sono state sostituite automaticamente dall' Assembler.
Nel secondo listato diremo che le macro sono state espanse.
La differenza concettuale tra una macro e una subroutine è evidente.
Una subroutine viene scritta una sola volta in memoria e interrompe il flusso lineare di un programma in quanto si esegue un salto, mentre una macro viene scritta in memoria tutte le volte che viene chiamata e non interrompe il flusso normale di un programma in quanto viene inserita in esso laddove è necessario.
Dunque non si usano le macro per risparmiare memoria, ma per risparmiare il tempo necessario a riscrivere spezzoni simili di un programma e per realizzare delle "super istruzioni", delle macro-istruzioni appunto.
Come copiare un programma in linguaggio macchina?
Normalmente nelle riviste viene pubblicato un listing tipo il listing 2, cioè con il programma oggetto accanto al programma sorgente. Talvolta tuttavia viene pubblicato soltanto il programma oggetto e questa è una pessima abitudine in quanto il programma oggetto è in sè e per sè totalmente inintelligibile e se un povero lettore vuol capirci qualcosa deve prima copiarlo nel computer e poi disassemblarlo con il Disassembler.
Io consiglio quindi a coloro che intendono pubblicare programmi in linguaggio macchina su riviste di far stampare sia il programma sorgente che quello oggetto e di abbondare nei commenti.
Vediamo comunque come si ricopia un programma in linguaggio macchina tratto da una rivista.
1) Se si dispone di un Assembler e il sorgente è stato pubblicato, conviene copiarlo col proprio Assembler in modo da poterlo modificare secondo le proprie esigenze o semplicemente rilocarlo in altra zona di memoria.
2) Se non si dispone di un Assembler o se il programma sorgente non è stato pubblicato si ricopia il programma oggetto direttamente dal monitor dell' Apple. In sostanza per ricopiare l' Amper-Interpreter basta che introduciate dal monitor i numeri esadecimali della seconda colonna del listato 2.
Per introdurre i numeri esadecimali bisogna prima di tutto fare: *CALL - 151* per entrare nel monitor, poi prendere la locazione di partenza del programma in linguaggio macchina (nel nostro caso *9900*), scriverla sul video e mettere i due punti.
Poi si comincia a scrivere i valori esadecimali uno dietro l'altro separandoli con uno spazio. Alla fine si preme il tasto *RETURN*.
Per minimizzare il tempo richiesto da questa noiosissima operazione di copia si può usare un registratore a cassette e registrare la sequenza di numeri leggendoli a voce alta. Poi si riascoltano i numeri dal registratore e si confrontano con quelli stampati sulla rivista in modo da controllarne la correttezza. Fatto questo. si riascoltano i numeri registrati e si digitano sul computer, quindi si riascoltano e se ne controlla la correttezza.
Si consiglia di prestare attenzione alle D che si possono confondere con gli 0 e gli 8 che assomigliano alle B.
Per chi non dispone dell' S-C Macro Assembler, ma di altri Assembler spiego il significato delle pseudo più peculiari di questo Assembler.
.DA serve a scrivere degli indirizzi di memoria secondo il formato tipico del 6502 (byte basso prima, byte alto dopo)
.N (*n*=numero) sono etichette locali e sono definite internamente in relazione all'etichetta normale che viene prima di loro. In sostanza:

```
PRIMO      STA  $00
.1         LDA  $00
           BNE  .1   va in su
SECONDO    STA  $00
           BNE  .1   va in giù
.1         LDA  $00
```

perchè il significato di .1 è "locale" allo spezzone di programma fra due etichette consecutive.
Queste etichette si possono sostituire con altre etichette normali prestando attenzione però al loro significato intrinseco (dare cioè due nomi diversi a due etichette .1 che vengono dopo due etichette normali diverse).
.AS serve a scrivere *una* stringa di caratteri in memoria. Tutti i byte più significativi dei caratteri sono a 0.

.*HS* serve a scrivere un numero esadecimale in memoria.
.*TF* nome file = transfer file: serve per assemblare il sorgente su disco in un file specificato.

Il comando &

Vediamo ora la teoria dell' Amper-Interpreter.
Cosa succede quando l'interprete BASIC incontra un & in un programma?
La cosa è illustrata nella figura 1 dove si vede che il BASIC salta (*per l'esattezza, attraverso una chiamata di subroutine, ossia un comando JSR.* NdR) alla locazione *$3F5* dove incontra un' istruzione *JMP* (salto) alla locazione di memoria indicata nel byte *$3F6* (parte bassa) e *$3F7* (parte alta).
In queste due locazione , che si trovano nella RAM l'utente può scrivere l'indirizzo di partenza del suo programma di gestione del comando ampersand .
Si tratta dunque di "un' istruzione aperta" , cioè di un'istruzione lasciata in bianco e che può essere utilizzata dall'utente a suo piacere.
Ovviamente nel nostro caso le due locazione di memoria punteranno all' Amper-Interpreter, l' Amper-Interpreter viene così agganciato al BASIC e ne diventa a tutti gli effetti un'estensione.
Vorrei sottolineare che questa possibilità dell'Applesoft di venir esteso è dovuta al fatto che esiste un pezzettino di BASIC (alle locazione *$3F5*, *$3F6* e *$3F7*) che si trova in RAM e può quindi venir facilmente modificato.
Vi domanderete "cosa succede quando spengo l' Apple?".
Evidentemente questi tre byte vanno persi, per cui l'Applesoft ,quando viene acceso il computer (se non contiene routine di gestione del comando ampersand), provvede a scrivere in questa locazione l'istruzione JMP e l'indirizzo "di default" della routine del comando ampersand .
Questo indirizzo è *$FF58* e a questa locazione di memoria si trova l'istruzione *RTS*, cioè l'istruzione di ritorno dalla subroutine (ne più ne meno).
In conclusione cosa succede? Niente.
Ben diverso è il nostro caso poichè noi andiamo direttamente a modificare l'indirizzo di destinazione del salto e costringiamo il BASIC ad agganciare l'Amper-Interpreter.

Caratteristiche dell' Amper-Interpreter

Esistono programmi come l'APA e il Renumber che riconoscono solo l'iniziale del comando, altri come il Softplus e il XBASIC che accettano solo parole chiave del BASIC (come *PRINT* e *INPUT* ad es.), l' Amper-Interpreter invece cosa riconosce? Tutto!
L'Amper-Interpreter accetta comandi qualsiasi formati da un numero x di lettere e tutte sono significative, non solo la prima.L'Amper-Interpreter accetta parole chiave Applesoft e le tratta in modo corretto.
L' Amper-Interpreter accetta miscugli di parole chiave e caratteri qualsiasi, ad esempio *PRINTPAGE* insomma non pone alcun limite.
Sulla teoria dei comandi tratteremo nel prossimo articolo, ma già da adesso si possono utilizzare 4 nuovi e comodi comandi:& < permette di avere solo 33 colonne sul video e serve per i listing sul monitor. & > ripristina le 40 colonne su video. *& MNTR* è equivalente ad un *CALL -151* e serve per saltare al MoNiTR.
&L è equivalente ad un *LIST* (verrà potenziato in seguito).
Con il primo, il secondo e il quarto comando si potranno dunque editare e visualizzare più rapidamente i programmi. Con il terzo si avrà un rapido accesso al monitor dell'Apple in quanto le lettere *M, N* e *T, R* sono adiacenti e si battono rapidamente.
Se si vuol cambiare il nome dei comandi, lo si può fare facilmente con un Assembler .
Non consiglio di ridurre tutti i comandi ad una singola lettera perchè questo sistema genera facilmente confusione e dopotutto le lettere dell'alfabeto sono relativamente poche, quindi solo pochi comandi si possono distinguere.
Si consiglia di usare una o poche lettere coi comandi veramente essenziali e di uso frequente.
La figura 2 mostra la struttura dell'Amper-Interpreter.
Alle locazione di memoria più basse si trova l'interprete vero e proprio che analizza l'istruzione (da noi inventata) nella linea BASIC e controlla che essa sia una di quelle supportate dall'Amper-Interpreter.
La lista dei comandi supportati si trova nella zona denominata Command Table . In caso di riconoscimento il controllo è passato all'opportuna routine (zona Command Routine) che gestisce il particolare comando.
Se invece il comando non è tra quelli riconosciuti dall'Amper-Interpreter allora il controllo viene passato ad un'eventuale altra routine che usa l' ampersand o, in mancanza di questa, alla routine che genera il *SYNTAX ERROR*.
E' possibile pertanto collegarsi per esempio all' APA.
L'interfacciamento dell'Amper-Interpreter con altri programmi verrà comunque visto in seguito.
La parte indicata con Address Table nella figura 2 contiene gli indirizzi delle routine che gestiscono i vari comandi riconosciuti.
La figura 3 mostra il flow chart dell'Amper-Interpreter.
Come si vede il cuore del programma è la routine *COMPARE* che confronta la stringa nella linea di programma con i comandi riconosciuti. Se viene riconosciuto un comando viene settato il carry, in caso contrario viene azzerato.
Comunque procediamo con ordine.
La parte di *inizializzazione* provvede a collegare il programma al BASIC e a collegarsi con la routine di *SYNTAX ERROR* nel caso il comando non fosse riconosciuto. Il programma comincia a *START* mettendo a *$FF* (-1 decimale) il contatore dei comandi riconosciuti dall' Amper-Interpreter che sono stati confrontati con quello che si trova nella linea BASIC.
Poi si controlla che la tabella dei comandi non sia terminata. Questo si ottiene controllando la parte alta dell'indirizzo della routine del prossimo comando. Se questa è nulla allora vuol dire che la lista dei comandi è finita. Se tutto va bene si passa alla subroutine *COMPARE* e se al suo ritorno il carry è settato il controllo è passato all' apposita routine che lo gestisce.
Maggiori dettagli su come questo sia ottenuto verranno descritte nel prossimo articolo dove queste conoscenze sono necessarie per la comprensione dei comandi che verranno introdotti.
Mi preme di più adesso spiegare come si fa a fondere l'Amper-Interpreter con l'ambiente BASIC più DOS onde evitare fenomeni di "rigetto" ed ottenere una perfetta elasticità di impiego.

Fusione con il DOS

È assolutamente necessario che l'Amper-Interpreter si trovi in un luogo sicuro, al riparo da accidentali sovrascritture e cancellazioni da parte del BASIC o di altri programmi.
Esiste una regione della memoria dell'Apple che ha appunto questa prerogativa: la zona fra il DOS e i suoi buffer.
La figura 4 mostra la parte di memoria contenente il DOS

e i buffer. Questi ultimi sono generalmente in numero di tre, ma questa cifra può essere modificata con il comando Maxfile (questo però non c'entra proprio niente col nostro discorso).
Cosa sono i buffer?
I buffer sono delle zone di memoria sotto il DOS lunghe 595 byte l'una e servono per l'input e l'output con i dischi.
Himem, cioè l'indicatore BASIC della massima locazione di memoria disponibile all'Applesoft per la memorizzazione di programmi e variabili, si trova sempre sotto i buffer con il risultato quindi che per l'Applesoft la zona di memoria buffer più DOS è tabù o, se preferite, è come se non esistesse.
Mettendo quindi l'Amper-Interpreter fra il DOS e i buffer siamo sicuri che all'Applesoft non verrà mai in mente di scriverci sopra (figura 5).
Rimane il problema di come creare questo spazio fra il DOS e i buffer.
I buffer sono localizzati da un puntatore posto nelle locazione di memoria *$9D00* e *$9D01 (40192, 40193)*.
Noi dobbiamo soltanto diminuire questo puntatore del numero di byte che vogliamo inserire tra il DOS e i buffer, inserire il programma, e poi chiamare la subroutine alla locazione *$7D5 (42964)* che ricostruisce i buffer a partire dalla locazione indicata in *$9D00* e *$9D01*. La procedura completa di inizializzazione di un disco contenente l'Amper-Interpreter verrà descritta in seguito, dopo cioè che avremo trattato un altro problema, che per adesso non dico.

Collegamento al BASIC

Fino ad ora abbiamo visto come inserire l'Amper-Interpreter in modo che non venga cancellato.Vedremo adesso come fare per collegarlo al BASIC in modo totalmente trasparente all'utilizzatore,in maniera tale cioè che al momento del boot l'Amper-Interpreter venga inizializzato automaticamente senza bisogno di lanciare un programma di *HELLO* o cose del genere.
Insomma si farà in modo che l'Amper-Interpreter divenga in tutto e per tutto un'estensione del BASIC Applesoft presente immediatamente quando il computer viene acceso o quando viene fatto un *PR#6* con il disco che lo contiene.Per ottenere ciò bisogna modificare il DOS opportunamente.
Le modifiche sono riportate nel listato 3 che mostra la piccola "pezza" da attaccare al DOS affinchè metta nelle locazioni *$3F6* e *$3F7* l'indirizzo dell'Amper-Interpreter.
Attenzione che ci sono in effetti 2 "pezze" (patches è il termine anglosassone), una alla locazione *$B6B3* e seguenti, una alla locazione *$B6E8* e seguenti.
I due patches si chiamano *DOS1* e *DOS2*.
Se non si usa un Assembler si faccia attenzione agli indirizzi di memoria da modificare.Abbiamo così inserito in una zona non utilizzata del DOS (*$B6B3* in poi e *$B6E8* in poi) una routine di controllo sulle locazioni *$3F6* e *$3F7* e un allacciamento alle eventuali altre routine che utilizzano il comando &.
Supponiamo ad es. di volerci collegare con l'APA del DOS Tool Kit.
Allora dobbiamo:

1) caricare il disco con l'AMI;
2) fare *RUN LOADAPA* (a questo punto l'Amper-Interpreter è sconnesso , mentre l'APA è connesso);
3) fare (dal monitor) *3D0G*, oppure (dal BASIC) *CALL 976*.

In sostanza se per un qualsiasi motivo l'Amper-Interpreter è sconnesso dal BASIC , basta fare un warm-start del DOS ,cioè una partenza a "caldo", cosa che si ottiene chiamando la routine alla locazione *$3D0 (976)*.

Inizializzare un disco

La figura 6 mostra la procedura completa da seguire per inizializzare un disco contenente l'Amper-Interpreter.
Le prime operazioni servono alla creazione di uno spazio tra il DOS e i buffer ,le successive ad agganciarci con l'Amper-Interpreter e ad inizializzare un dischetto con il DOS modificato.
Il DOS sarà modificato per tre ragioni:
1) deve lasciare uno spazio tra sè e i buffer per contenere l'Amper-Interpreter;
2) deve inglobare nello spazio creato l'Amper-Interpreter come parte di sè;
3) deve inglobare i programmi *DOS1* e *DOS2* al suo interno.
Ciò e ' stato ottenuto cambiando opportuni puntatori all'interno del DOS stesso.
Quando si fa *PR#6* o si accende il computer con questo nuovo disco dentro, si ottiene immediatamente un BASIC esteso, comprendente cioè le istruzioni dell'Amper-Interpreter ,il quale sarà già completamente inizializzato e funzionante.

Tirando le somme, per copiare l'Amper-Interpreter si devono fare i seguenti passi:
Per chi usa un Assembler.
1) Editare il programma con un Assembler e assemblarlo su disco.
2) Editare *DOS1* e *DOS2* con un Assembler e assemblarli su disco (due file separate, mi raccomando)
3) Seguire le istruzioni di figura 6.
Per chi non usa un Assembler .
1) Copiare direttamente in memoria il programma oggetto dalla locazione *$9900* in poi facendo *CALL-151 <ret> 9900: A9 13 8D ....*
2) Copiare direttamente in memoria i programmi *DOS1* e *DOS2* facendo *DOS1 B6B3:Ad B6 AA ..... DOS2 B6E8: A9 99 A0 13 .....*
3) Seguire le istruzioni di figura 6 saltando ovviamente i *BLOAD* ivi descritti.
Nei prossimi numeri vedremo in dettaglio come funziona l'Amper-Interpreter e le routine che usa e introdurremo numerosi altri comandi per l'editing, la grafica, il linguaggio macchina, e l'animazione.

```
:LIST

1000 *-----------------------------------
1010 *        >INCD P1      INCREMENT 16-BIT VALUE
1020 *-----------------------------------
1030          .MA INCD
1040          INC ]1
1050          BNE :1
1060          INC ]1+1
1070 :1
1080          .EM
1090 *-----------------------------------
1100 *        >PNTR P1,P2   BUILD A POINTER TO P1
1110 *                      (PUTS ADDRESS OF P1 AT P2,P2+1)
1120 *-----------------------------------
1130          .MA PNTR
1140          LDA #]1
1150          STA ]2
1160          LDA /]1
1170          STA ]2+1
1180          .EM
1190 *-----------------------------------
1200          .OR $9900
1210          .TF AMP.CODE
1220 TXT.PTR .EQ $B8       CHARGET PTR
1230 AMPVECT .EQ $3F6      & $3F7
1240 CONT    .EQ $F9
1250 NEXT.TBL .EQ $FA      & $FB
1260 COMM.PTR .EQ $FE      & $FF
1270 RETURN .EQ $0D        CODICE HEX DEL RETURN
1280 *-----------------------------------
1290 CHARGET     .EQ $B1   PRENDE IL PROSSIMO CHAR NELLA LINEA
1300 LIST    .EQ $D6A5     LIST
1310 SYNERR .EQ $DEC9      SYNTAX ERROR
1320 DATA    .EQ $D995     CERCA FINE LINEA BASIC
1330 MONZ    .EQ $FF69     ENTRATA AL MONITOR COMMAND PROCESSOR
1340 *-----------------------------------
1350 INIT    >PNTR START,AMPVECT
1360         >PNTR SYNERR,NEXT.TBL
1370         RTS
1380 *-----------------------------------
1390 START   LDX #$FF      X=-1
1400         STX CONT      CONT=-1
1410         >PNTR COMM.TBL-1,COMM.PTR
1420 MAIN.LOOP
1430         INC CONT      AGGIORNA CONT
1440         LDA CONT      CARICALO
1450         ASL           RADDOPPIALO
1460         TAY           METTLIO IN Y
1470         INY           INCREMANTALO
1480         LDA AD.TBL,Y CARICA BYTE ALTO DELLA ROUTINE
1490         BEQ NOT.IN.TBL SE 0 ALLORA FINE TABELLA QUINDI SALTA
1500         JSR COMPARE  CERCA COMANDO
1510         BCC MAIN.LOOP C=0 UGUALE NON C'E',ALLORA SALTA
1520 *-----------------------------------
1530         LDA CONT      CARICA CONT
1540         ASL           RADDOPPIALO
1550         TAY           METTILO IN Y
1560 *-----------------------------------
```

```
1570 * CARICA INDIRIZZO DI ROUTINE
1580 * IN STACK E RTS PER SALTARCI
1590        LDA AD.TBL+1,Y
1600        PHA
1610        LDA AD.TBL,Y
1620        PHA
1630        RTS
1640 *-------------------------------
1650 COMPARE
1660        LDY #$FF       Y=-1
1670        LDX #$00       X=0
1680 NEXT.CHAR
1690        INY            AGGIORNO Y
1700        >INCD COMM.PTR
1710        LDA (COMM.PTR,X) CARICO UN CAR. DEL COMANDO DALLA TABELLA
1720        CMP #RETURN
1730        BEQ EXIT
1740        CMP (TXT.PTR),Y LO CONFRONTO CON LA LINEA BASIC
1750        BNE NEXT.COMM SE NON UGUALI SALTO
1760        BEQ NEXT.CHAR
1770 EXIT   SEC
1780        RTS
1790 NEXT.COMM
1800 SCAN
1810        >INCD COMM.PTR
1820        LDA (COMM.PTR,X) CARICO UN CAR. DEL COMANDO DALLA TABELLA
1830        CMP #RETURN
1840        BEQ .1
1850        JMP SCAN
1860 .1     CLC
1870        RTS
1880 *-------------------------------
1890 NOT.IN.TBL
1900        JMP (NEXT.TBL)
1910 *-------------------------------
1920 END.OF.COMM
1930        JSR DATA
1940        RTS
1950 *-------------------------------
1960 AMP.ROUTINES
1970 *-------------------------------
1980 COMM.SMALL
1990 * ESEGUE POKE 33,33
2000        LDA #$21
2010        STA $21
2020        JMP END.OF.COMM
2030 *-------------------------------
2040 COMM.LARGE
2050 * ESEGUE POKE 33,40
2060        LDA #$28
2070        STA $21
2080        JMP END.OF.COMM
2090 *-------------------------------
2100 COMM.MNTR
2110        JSR MONZ
2120 *-------------------------------
2130 COMM.LIST
2140 * PRENDE PROSSIMO CARATTERE E SALTA AL LIST DEL BASIC
2150        JSR CHARGET
2160        JMP LIST
2170 *-------------------------------
2180 * NUOVI COMANDI DA INSERIRE
2190 * NOTATE IL VUOTO NEI NUMERI DI LINEA
2200 *-------------------------------
2210 AD.TBL
2220        .DA COMM.SMALL-1,COMM.LARGE-1,COMM.MNTR-1,COMM.LIST-1,$000
2230 *-------------------------------
2240 COMM.TBL
2250        .HS D1         '<' TOKEN
2260        .HS OD
2270        .HS CF         '>' TOKEN
2280        .HS OD
2290        .AS 'MNTR'
2300        .HS OD
2310        .AS 'L'
2320        .HS OD
:PR#0
:ASM
```

Listato 1 - Programma sorgente.

```
                1000 *-------------------------------
                1010 *       >INCD P1     INCREMENT 16-BIT VALUE
                1020 *-------------------------------
                1030         .MA INCD
                1040         INC ]1
                1050         BNE :1
                1060         INC ]1+1
                1070 :1
                1080         .EM
                1090 *-------------------------------
                1100 *       >PNTR P1,P2  BUILD A POINTER TO P1
                1110 *                    (PUTS ADDRESS OF P1 AT P2,P2+1)
                1120 *-------------------------------
                1130         .MA PNTR
                1140         LDA #]1
                1150         STA ]2
                1160         LDA /]1
                1170         STA ]2+1
                1180         .EM
                1190 *-------------------------------
                1200         .OR $9900
                1210         .TF AMP.CODE
00B8-           1220 TXT.PTR .EQ $B8     CHARGET PTR
03F6-           1230 AMPVECT .EQ $3F6    & $3F7
00F9-           1240 CONT    .EQ $F9
00FA-           1250 NEXT.TBL .EQ $FA    & $FB
00FE-           1260 COMM.PTR .EQ $FE    & $FF
000D-           1270 RETURN .EQ $0D      CODICE HEX DEL RETURN
                1280 *-------------------------------
00B1-           1290 CHARGET    .EQ $B1  PRENDE IL PROSSIMO CHAR NELLA LINEA
D6A5-           1300 LIST    .EQ $D6A5   LIST
DEC9-           1310 SYNERR .EQ $DEC9    SYNTAX ERROR
D995-           1320 DATA    .EQ $D995   CERCA FINE LINEA BASIC
FF69-           1330 MONZ    .EQ $FF69   ENTRATA AL MONITOR COMMAND PROCESSOR
                1340 *-------------------------------
9900-           1350 INIT    >PNTR START,AMPVECT
9900- A9 13     0000>         LDA #START
9902- 8D F6 03  0000>         STA AMPVECT
9905- A9 99     0000>         LDA /START
9907- 8D F7 03  0000>         STA AMPVECT+1
990A-           1360         >PNTR SYNERR,NEXT.TBL
990A- A9 C9     0000>         LDA #SYNERR
990C- 85 FA     0000>         STA NEXT.TBL
990E- A9 DE     0000>         LDA /SYNERR
9910- 85 FB     0000>         STA NEXT.TBL+1
9912- 60        1370         RTS
                1380 *-------------------------------
9913- A2 FF     1390 START   LDX #$FF     X=-1
9915- 86 F9     1400         STX CONT     CONT=-1
9917-           1410         >PNTR COMM.TBL-1,COMM.PTR
9917- A9 8E     0000>         LDA #COMM.TBL-1
9919- 85 FE     0000>         STA COMM.PTR
991B- A9 99     0000>         LDA /COMM.TBL-1
991D- 85 FF     0000>         STA COMM.PTR+1
                1420 MAIN.LOOP
991F- E6 F9     1430         INC CONT     AGGIORNA CONT
9921- A5 F9     1440         LDA CONT     CARICALO
9923- 0A        1450         ASL          RADDOPPIALO
9924- A8        1460         TAY          METTLIO IN Y
9925- C8        1470         INY          INCREMANTALO
9926- B9 85 99  1480         LDA AD.TBL,Y CARICA BYTE ALTO DELLA ROUTINE
9929- F0 3C     1490         BEQ NOT.IN.TBL SE O ALLORA FINE TABELLA QUINDI SALTA
992B- 20 3D 99  1500         JSR COMPARE  CERCA COMANDO
992E- 90 EF     1510         BCC MAIN.LOOP C=O UGUALE NON C'E',ALLORA SALTA
                1520 *-------------------------------
9930- A5 F9     1530         LDA CONT     CARICA CONT
9932- 0A        1540         ASL          RADDOPPIALO
9933- A8        1550         TAY          METTILO IN Y
                1560 *-------------------------------
                1570 * CARICA INDIRIZZO DI ROUTINE
                1580 * IN STACK E RTS PER SALTARCI
```

```
9934- B9 86 99  1590         LDA AD.TBL+1,Y
9937- 48        1600         PHA
9938- B9 85 99  1610         LDA AD.TBL,Y
993B- 48        1620         PHA
993C- 60        1630         RTS
                1640 *-------------------------------
                1650 COMPARE
993D- A0 FF     1660         LDY #$FF      Y=-1
993F- A2 00     1670         LDX #$00      X=0
                1680 NEXT.CHAR
9941- C8        1690         INY           AGGIORNO Y
9942-           1700         >INCD COMM.PTR
9942- E6 FE     0000>         INC COMM.PTR
9944- D0 02     0000>         BNE :1
9946- E6 FF     0000>         INC COMM.PTR+1
                0000> :1
9948- A1 FE     1710         LDA (COMM.PTR,X) CARICO UN CAR. DEL COMANDO DALLA TABELLA
994A- C9 0D     1720         CMP #RETURN
994C- F0 06     1730         BEQ EXIT
994E- D1 B8     1740         CMP (TXT.PTR),Y LO CONFRONTO CON LA LINEA BASIC
9950- D0 04     1750         BNE NEXT.COMM SE NON UGUALI SALTO
9952- F0 ED     1760         BEQ NEXT.CHAR
9954- 38        1770 EXIT    SEC
9955- 60        1780         RTS
                1790 NEXT.COMM
                1800 SCAN
9956-           1810         >INCD COMM.PTR
9956- E6 FE     0000>         INC COMM.PTR
9958- D0 02     0000>         BNE :1
995A- E6 FF     0000>         INC COMM.PTR+1
                0000> :1
995C- A1 FE     1820         LDA (COMM.PTR,X) CARICO UN CAR. DEL COMANDO DALLA TABELLA
995E- C9 0D     1830         CMP #RETURN
9960- F0 03     1840         BEQ .1
9962- 4C 56 99  1850         JMP SCAN
9965- 18        1860 .1      CLC
9966- 60        1870         RTS
                1880 *-------------------------------
                1890 NOT.IN.TBL
9967- 6C FA 00  1900         JMP (NEXT.TBL)
                1910 *-------------------------------
                1920 END.OF.COMM
996A- 20 95 D9  1930         JSR DATA
996D- 60        1940         RTS
                1950 *-------------------------------
                1960 AMP.ROUTINES
                1970 *-------------------------------
                1980 COMM.SMALL
                1990 * ESEGUE POKE 33,33
996E- A9 21     2000         LDA #$21
9970- 85 21     2010         STA $21
9972- 4C 6A 99  2020         JMP END.OF.COMM
                2030 *-------------------------------
                2040 COMM.LARGE
                2050 * ESEGUE POKE 33,40
9975- A9 28     2060         LDA #$28
9977- 85 21     2070         STA $21
9979- 4C 6A 99  2080         JMP END.OF.COMM
                2090 *-------------------------------
                2100 COMM.MNTR
997C- 20 69 FF  2110         JSR MONZ
                2120 *-------------------------------
                2130 COMM.LIST
                2140 * PRENDE PROSSIMO CARATTERE E SALTA AL LIST DEL BASIC
997F- 20 B1 00  2150         JSR CHARGET
9982- 4C A5 D6  2160         JMP LIST
                2170 *-------------------------------
                2180 * NUOVI COMANDI DA INSERIRE
                2190 * NOTATE IL VUOTO NEI NUMERI DI LINEA
                2200 *-------------------------------
                2210 AD.TBL
9985- 6D 99 74
9988- 99 7B 99
998B- 7E 99 00
998E- 00        2220         .DA COMM.SMALL-1,COMM.LARGE-1,COMM.MNTR-1,COMM.LIST-1,$000
                2230 *-------------------------------
                2240 COMM.TBL
998F- D1        2250         .HS D1        '<' TOKEN
9990- 0D        2260         .HS OD
9991- CF        2270         .HS CF        '>' TOKEN
9992- 0D        2280         .HS OD
9993- 4D 4E 54
9996- 52        2290         .AS 'MNTR'
9997- 0D        2300         .HS OD
9998- 4C        2310         .AS 'L'
9999- 0D        2320         .HS OD

SYMBOL TABLE

9985- AD.TBL
996E- AMP.ROUTINES
03F6- AMPVECT
00B1- CHARGET
9975- COMM.LARGE
997F- COMM.LIST
997C- COMM.MNTR
00FE- COMM.PTR
996E- COMM.SMALL
998F- COMM.TBL
993D- COMPARE
00F9- CONT
D995- DATA
996A- END.OF.COMM
9954- EXIT
9900- INIT
D6A5- LIST
991F- MAIN.LOOP
FF69- MONZ
9941- NEXT.CHAR
9956- NEXT.COMM
00FA- NEXT.TBL
9967- NOT.IN.TBL
000D- RETURN
9956- SCAN
.01=9965
9913- START
DEC9- SYNERR
00B8- TXT.PTR

0000 ERRORS IN ASSEMBLY
:PR#0
:ASM
```

Listato 2 - Programma oggetto più sorgente.

```
9900-           1000 INIT    .EQ $9900
                1010 *-------------------------------
                1020         .OR $B6B3
                1030         .TF DOS1.CODE
                1040 DOS1
                1050 *-------------------------------
B6B3- AD B6 AA  1060         LDA $AAB6    CONTROLLA SE
B6B6- 49 40     1070         EOR #$40     SIAMO IN APPLESOFT
B6B8- D0 3E     1080         BNE CONTINUE SE NO, COME NON DETTO
B6BA- B0 39     1090         BCS COLDSTART C=1 UGUALE COLDSTART
B6BC- AD F7 03  1100         LDA $3F7     CARICA & VECT (BYTE ALTO)
B6BF- C9 99     1110         CMP #$99     E'ATTACCATO L' AMP-INT?
B6C1- F0 35     1120         BEQ CONTINUE SI,COME NON DETTO
B6C3- AC F6 03  1130         LDY $3F6     CARICA IL NUOVO
B6C6- AD F7 03  1140         LDA $3F7     VETTORE &
B6C9- 84 FA     1150         STY $FA      METTILO IN
B6CB- 85 FB     1160         STA $FB      NEXT.TBL
B6CD- 4C E8 B6  1170         JMP DOS2     SALTA PERCHE' QUI NON C'E' PIU' SPAZIO
                1180 *-------------------------------
                1190 *-------------------------------
                1200         .OR $B6E8
                1210         .TF DOS2.CODE
                1220 DOS2
                1230 *-------------------------------
                1240 * CARICA L'INIZIO DELL' AMP-INT NEL VETTORE &
B6E8- A9 99     1250         LDA #$99
B6EA- A0 13     1260         LDY #$13
B6EC- 8D F7 03  1270         STA $3F7
B6EF- 8C F6 03  1280         STY $3F6
B6F2- 4C F8 B6  1290         JMP CONTINUE SALTA A CONTINUE
                1300 COLDSTART
                1310 * QUESTO E' UN BOOTSTRAP
B6F5- 20 00 99  1320         JSR INIT     INIZIALIZZA L' AMP-INT
                1330 CONTINUE
B6F8- 4C 51 A8  1340         JMP $A851     RIENTRA AL DOS NORMALE

SYMBOL TABLE

B6F5- COLDSTART
B6F8- CONTINUE
B6B3- DOS1
B6E8- DOS2
9900- INIT

0000 ERRORS IN ASSEMBLY
:PR#0
```

Listato 3 - Programma per inserire AMI.

Caricamento di routine da disco

Merge di routine e dotazione dei file: due problemi risolti su Apple

di **M. Cerofolini** e **V. Cuoghi**

Nelle riviste specializzate e in vari testi, appaiono spesso routine o programmi che possono essere inseriti come subroutine all'interno di un proprio lavoro.

E' quindi senz'altro utile disporre di un metodo atto a:

a) caricare in memoria una subroutine registrata precedentemente su disco;

b) fonderla con un programma che già si trova in memoria.

Il metodo descritto in questo articolo non richiede programmi di utilità particolari all'infuori del normale *RENUMBER/APPEND* che viene fornito col disco sistema del DOS 3.3.

La routine che interessa deve essere salvata sotto formato *TEXT* con la numerazione delle linee BASIC a partire, ad esempio, dal numero 60.000.

Si potrà anche creare una piccola biblioteca di queste *routine* salvandole su un unico dischetto: basta avere l'accortezza di salvarle tutte con la stessa numerazione (per rimanere all'esempio precedente si tratterà sempre di 60.000).

La prima azione da compiere è il caricamento del programma *RENUMBER/APPEND* lanciandolo dal dischetto sistema DOS 3.3.

Quando nel corso della preparazione del programma principale si ha necessità di inserire in esso una delle routine predefinite, si esegua il comando:

EXEC NOME.ROUTINE

Esso carica in memoria le linee BASIC che costituiscono tale routine a partire dalla linea 60000, come sopra detto.

A questo punto, col *RENUMBER/APPEND* che in precedenza era stato caricato in memoria, si esegue il comando:

& Fxxxx,Iyy,S60000,E63999

La routine è adesso in memoria insieme al programma che si stava sviluppando con le righe rinumerate a partire dal valore "xxxx" con incremento "yy". Con lo stesso sistema si possono fondere più routine in un unico programma.

Per salvare sotto formato *TEXT* la routine si può aggiungere alla routine stessa una riga con numero 0 (zero) fatta in questo modo:

0 PRINT CHR$(4); "OPEN NME. ROUTINE" : PRINT CH

Questa linea può a sua volta essere salvata sotto formato *TEXT* ad esempio col nome *LINEA.ZERO* e si può fondere con la routine da salvare in formato *TEXT* facendo

EXEC LINEA.ZERO.

Riassumendo il metodo descritto consiste delle seguenti operazioni.

1 Preparare il file *TEXT* contenente la linea zero usando il programma di figura 1 che richiede la digitazione della *LINEA.ZERO*. Questa deve essere digitata con molta attenzione così come è stata descritta in precedenza. Prima di dare *RETURN* si deve verificare l'esattezza di quanto scritto e, se si riscontra un errore, occorre ripetere tutta l'operazione. *Non si possono usare i normali tasti di editing.*

2 Scrivere o caricare da disco la routine (numeri di linea che partono da 60000).

3 Fare *EXEC LINEA.ZERO.* Questa operazione fonde con la nostra routine la linea zero vista in precedenza.

4 Fare il *RUN* del programma. Questa operazione salva sotto formato *TEXT* la routine in memoria con il nome *NOME.ROUTINE.*

5 Per dare il nome che si vuole alla routine, eseguire il comando DOS

RENAME NOME.ROUTINE, nome-effettivo.

Per fondere la routine col programma già esistente in memoria le operazioni sono le seguenti:

1 Se non lo si è fatto in precedenza, caricare la routine di *RENUMBER/APPEND* da dis-

DA BIT-ALADINO AL GENIO DELLA LAMPADA: CON RICEVUTA DI RITORNO

Tempo fa, in redazione, qualcuno lanciò l'idea di creare una bibliotechina di routine predefinite o standard che ciascun lettore potesse inserire nei propri "main" ogniqualvolta gli servisse, senza cioè necessità di reinventare "the wheel" come dicono gli anglofoni (o l'acqua calda, come diciamo noi latini ...). Ed ecco, schematizzato, un dialoghetto tra due **Bit**-redattori che, per non far nomi, identificheremo con le sigle D.Q. (Don Quixote) e S.P. (Sancho Panza):

D.Q. - *Sarebbe però bello poterlo fare in modo più o meno automatico, come avviene in moderni linguaggi di programmazione ...*

S.P. - *Pensieri oziosi d'una persona oziosa! Con le righe numerate del basichese è, a dir poco, un bel pasticcio.*

D.Q. - *Eppure vi sono utility di ri-numerazione talora incorporate nel DOS, ergo ...*

Il resto del dialogo, che solo a voler essere pomposi potremmo definire studio di fattibilità (in quanto alternava momenti costruttivi a fasi in cui l'epiteto più gentile era: *non-dire-idiozie*), vedeva una graduale conversione di S.P. all'utopia di D.Q., con crescente precisazione dei contorni tecnici proprio da parte del primo. A quel punto occorreva però un esperto autentico e meno oberato da orrendi compiti come rispondere alle lettere o fare il menabò o rincorrere novità senza quasi mai poterle approfondire.

Detto e fatto: si scrive a Modena al buon Matteo Cerofolino. Questi risponde abbastanza presto per promettere il proprio impegno, anche se prevede tempi lunghi.

E invece non passano più di venti giorni ed ecco il risultato, ottenuto con la collaborazione della moglie Valeria Cuoghi. L'esemplarità di questa storia non sfiora il deamicisismo? Può essere, però è un fatto, cui occorre aggiungere che i soft-coniugi modenesi riportano pure una modifica del DOS 3.3 dell'Apple che consente la datazione dei file in analogia a quanto offre l'UCSD Pascal.

Non ci resta che lanciare una sfida: perchè esperti di altri sistemi operativi non inviano qualcosa di analogo? Lo pubblicheremmo volentieri, anche nel caso fosse corredato delle sole modalità operative. Può darsi infine, che vi siano trucchi alternativi a quello escogitato dal duo Cuoghi-Cerofolini. Bene: dite la vostra che esso ha detto la sua.

co. Per fare questo occorre prima salvare il programma su cui si sta lavorando, dare *RUN RENUMBER/APPEND* e poi ricaricare il programma originario.
2. Fare la *EXEC* della routine che si vuole inserire in memoria. Essa sarà fusa insieme al programma.
3. Rinumerare la routine che si è caricata col comando *&Fxxxx,Iyy-,S60000,E63999.*
4. Ripetere, se necessario, l'operazione partendo dal punto 2.

La routine di datazione dei file

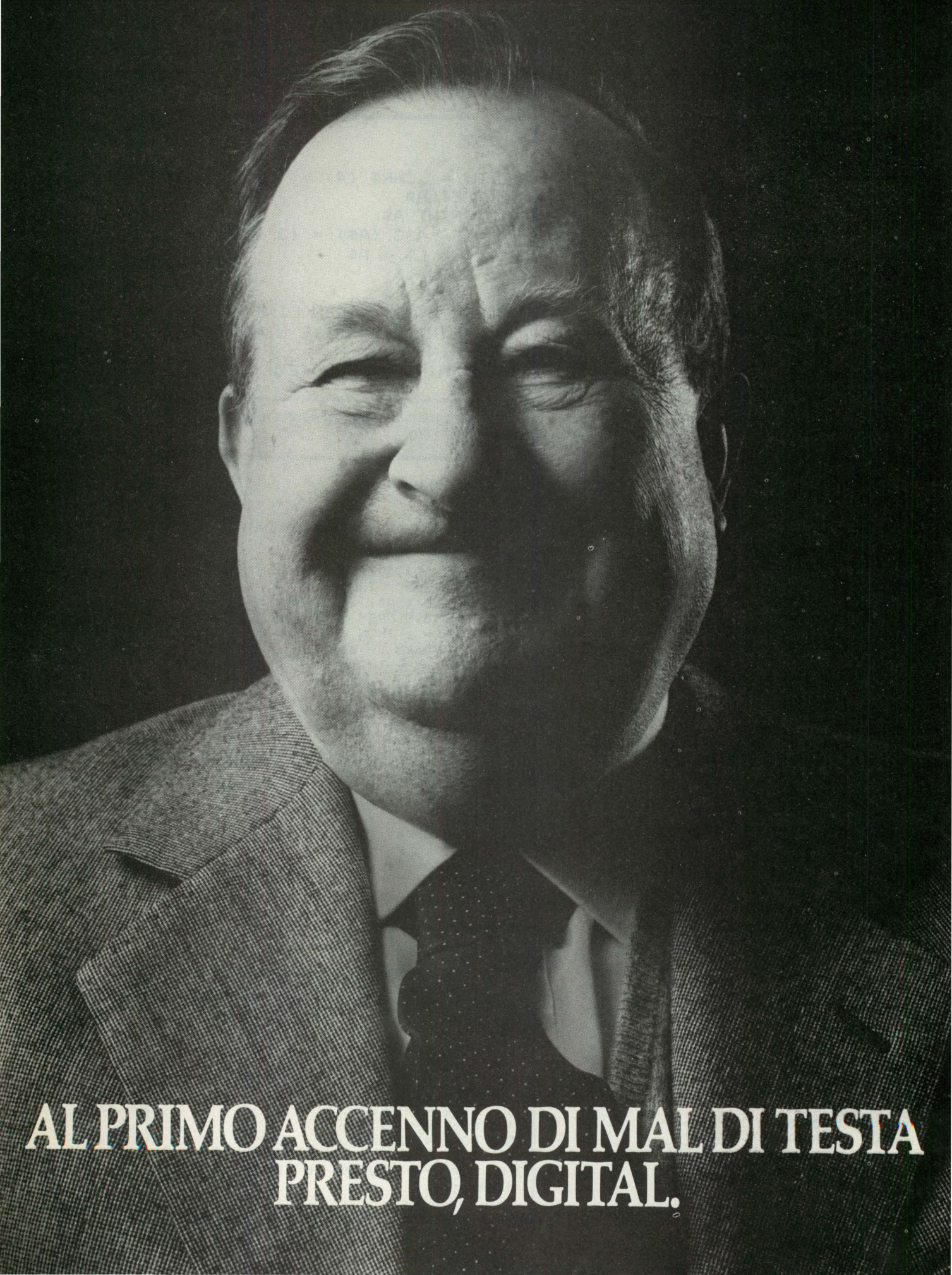
Il programma presentato in figura 2 permette di dotare il DOS 3.3 di una funzione di datazione simile a quella che si ha nel sistema operativo UCSD Pascal.
Usato al momento del boot come *HELLO*, il programma di figura 2 chiede, come prima cosa, la data del giorno. Se viene digitato qualcosa di diverso dal solo *RETURN* significa che si vuole usare la funzione di datazione dei file. Viene eseguito un controllo sulla data digitata e, se questa viene trovata formalmente corretta, viene inserita in una zona libera di memoria all'interno del DOS 3.3.
Subito dopo viene effettuata una patch del DOS nei comandi di *SAVE* e *BSAVE* affinchè in queste fasi inserisca, oltre al nome del file, anche la data, prelevata dalla zona di memoria dove era stata precedentemente salvata.
La figura 3 mostra la patch di cui si è detto. Il programma in assembler di figura 3 è *stato* riportato solo per illustrare il funzionamento della patch stessa e *non deve essere digitato* poichè è già compreso nel programma BASIC di figura 2.
Da questo momento, a tutti i file che verranno salvati tramite *SAVE* o *BSAVE*, verrà aggiunta, negli ultimi 8 caratteri, la data. La figura 4 è un esempio di un *CATALOG* di un dischetto contenente dei file datati ottenuti con questo metodo.

Un'applicazione

Come applicazione di quanto detto all'inizio sul caricamento di routine da disco e di quanto detto sulla patch di datazione dei file, presentiamo una routine (figura 5) che fornisce nella stringa *DT$* la data nel formato giorno-mese-anno, mentre nella stringa *TD$* la data nel formato anno-mese-giorno. Questa conversione di formato è utile tutte le volte che si debbono confrontare due date per vedere quale delle due è la maggiore.
Dopo essere stata salvata su dischetto, la routine di figura 5 può essere inserita in un qualsiasi programma facendone l'*EXEC*. Naturalmente questa routine è utile solo se si è proceduto ad inserire nel DOS 3.3 la patch che è stata descritta in precedenza. ■

```
]LIST

10 D$ =  CHR$ (4)
15  GET A$
20  PRINT A$;
30  IF  ASC (A$) = 13 THEN 60
40 L$ = L$ + A$
50  GOTO 15
60  PRINT L$
70  PRINT D$; "OPEN LINEA. ZERO"
80  PRINT D$; "WRITE LINEA. ZERO"
90  PRINT L$
100  PRINT D$; "CLOSE"
```

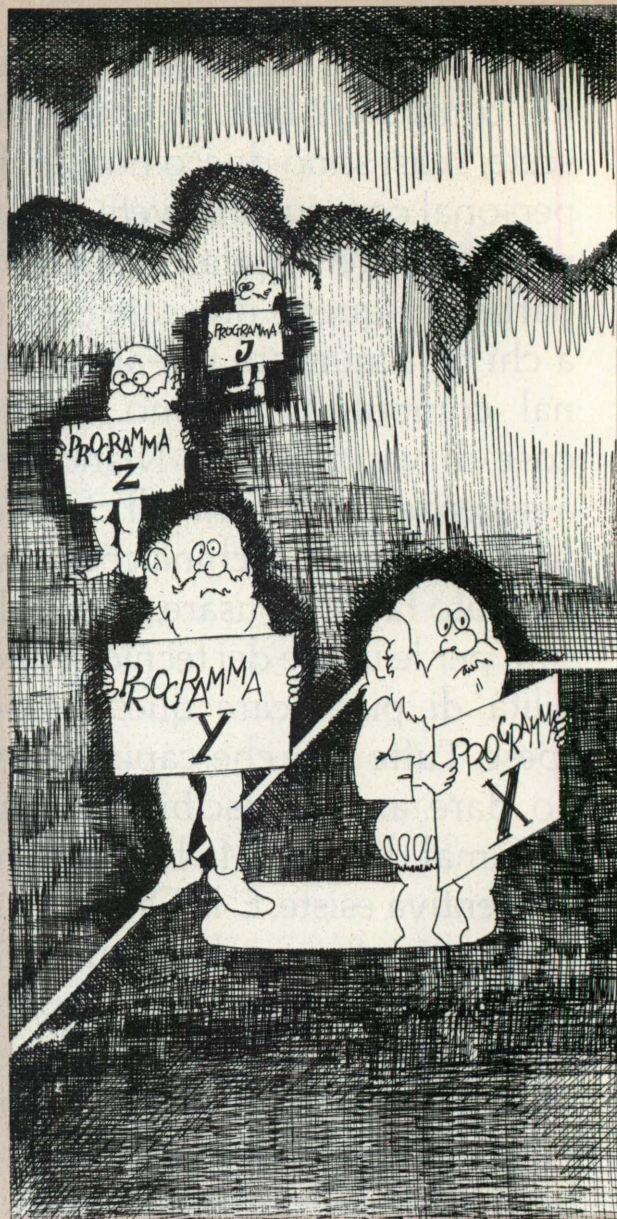
Figura 1 - Programmino per preparare il file **TEXT** *contenente la linea zero.*

```
]LIST

10  HOME
20  PRINT "-- PROGRAMMA DI DATAZIONE DE
I FILES --"
30  PRINT
40  INPUT "DATA? (GG/MM/AA) "; DA$
50  IF DA$ = "" THEN  END
60  IF  LEN (DA$) <  > 8 THEN 40
70  GOSUB 390: REM  CONTROLLO DATA
80  IF ER = 1 THEN 40
90  REM  $B6B3 -->46787
100  REM  ---- INSERISCE DATA NEL BUFFE
R ----
110  FOR I = 1 TO 8
120 A =  ASC ( MID$ (DA$, I, 1)) + 128
130  POKE 46787 - 1 + I, A
140  NEXT I
150  REM  ---- CARICA ROUTINE IN LINGUA
GGIO MACCHINA ---"
160  RESTORE
170 PA = 46771: REM  $B6C3
180  READ B
190  IF B = 999 THEN 230
200  POKE PA, B
210 PA = PA + 1
220  GOTO 180
230  REM  ---- INSERISCE JSR $B6B3 ----
------
240  REM  ---- PATCH PER COMANDO BSAVE
--------
250  POKE 41791, 32
260  POKE 41792, 179
270  POKE 41793, 182
280  REM  ---- PATCH PER COMANDO SAVE -
------
290  POKE 41893, 32
300  POKE 41894, 179
310  POKE 41895, 182
320  REM  ---- PATCH PER COMANDO SAVE -
------
330  POKE 41918, 32
340  POKE 41919, 179
350  POKE 41920, 182
360  END
370  REM  --- ROUTINE IN LINGUAGGIO MAC
CHINA ---
```

AL PRIMO ACCENNO DI MAL DI TESTA
PRESTO, DIGITAL.

Cariacamento di routine da disco

```
380  DATA  72,160,7,185,195,182,153,139
,170,136,16,247,104,76,213,163,999
390  REM  ---- ROUTINE DI CONTROLLO DAT
A ----
400 ER = 0
410 GG$ =  LEFT$ (DA$,2):MM$ =  MID$ (D
A$,4,2):AA$ =  RIGHT$ (DA$,2)
420 AA =  VAL (AA$):MM =  VAL (MM$):GG
=  VAL (GG$)
430  IF GG < 1 OR GG > 31 THEN ER = 1
440  IF MM < 1 OR MM > 12 THEN ER = 1
450  IF AA < 0 OR AA > 99 THEN ER = 1
460  RETURN
```

Figura 2 - Questo programma permette la datazione dei file come in UCSD Pascal.

```
1000
*-----------------------
---------
1002 *
1010 * PATCH PER
INSERIRE LA DATA
1020 * NEI PROGRAMMI
CHE VENGONO
1030 * SALVATI TRAMITE
I COMANDI
1032 *
1040 *          SAVE E
BSAVE
1042 *
1050 * AGLI INDIRIZZI
$A33F, $A3A5
1060 * E $A3BE
CORRISPONDENTI ALLE
1070 * ROUTINES BSAVE E
SAVE VA
1080 * INSERITA
L'ISTRUZIONE
1082 *
1090 *          JSR DATA
1100
*-----------------------
```

```
---------
                1180          .OR $B6B3
                BUCO NEL DOS
                1190          .TA $800
B6B3- 48        1200 DATA     PHA
SALVA 'A'
B6B4- A0 07     1210          LDY #7
B6B6- B9 C3 B6  1220    1     LDA
GG MM.AA,Y
B6B9- 99 8B AA  1230          STA $AA8B,Y
ULTIMI 8 CRT DEL NOME FILE
B6BC- 88        1240          DEY
```

```
B6BD- 10 F7      1250          BPL . 1
B6BF- 68         1260          PLA
B6C0- 4C D5 A3   1270          JMP $A3D5
                 1280
                 *----------------------
B6C3-            1290 GG. MM. AA . BS 8
SYMBOL TABLE
B6B3- DATA
 01=B6B6
B6C3- GG. MM. AA
0000 ERRORS IN ASSEMBLY
```

Figura 3 - Patch per inserire la data.

```
]CATALOG
DISK VOLUME 254
 A 002 HELLO
 T 004 NOME.ROUTINE
 T 002 LINEA.ZERO
 A 006 HELLO.DATA                 15/11/82
 A 003 CREA.LINEA.ZERO            15/11/82
 A 003 ROUTINE.DATA               16/11/82
```

Figura 4 - Esempio di ottenimento di file datati, col programma di figura 2.

```
]LIST
60000  REM  --------------------------
-
60001  REM  - QUESTA ROUTINE FORNISCE
-
60002  REM  - NEI CAMPI DT$ E TD$ LA
-
60003  REM  - DATA DIGITATA AL MOMENTO
-
60004  REM  - DEL BOOT
-
60005  REM  --------------------------
-
60010 DT$ =  CHR$ ( PEEK (46787)) +  CH
R$ ( PEEK (46788)) +  CHR$ ( PEEK (4679
0)) +  CHR$ ( PEEK (46791)) +  CHR$ ( P
EEK (46793)) +  CHR$ ( PEEK (46794))
60030 TD$ =  CHR$ ( PEEK (46793)) +  CH
R$ ( PEEK (46794)) +  CHR$ ( PEEK (4679
0)) +  CHR$ ( PEEK (46791)) +  CHR$ ( P
EEK (46787)) +  CHR$ ( PEEK (46788))
60050 RETURN
```

Figura 5 - Un caso applicativo di routine "di biblioteca".

CBM

Life: la vita continua

Simulazione dei processi vitali con il PET

di L. Pampana Biancheri

Sono certo che molti di noi già conoscono *Life*, il gioco ideato dal matematico John Norton Conway con l'intento di simulare i processi vitali in uno spazio a due dimensioni.
Le regole sono semplicissime. Preso un foglio quadrettato, si disegna una configurazione iniziale di cellule (ogni quadretto ne può contenere al massimo una). Poi si passa a determinare la prima "generazione": si calcola quali sono le cellule che sopravvivono, quali quelle che nascono e quelle che muoiono in base alle seguenti norme:

- una cellula nasce in ogni quadretto vuoto circondato (di lato o di spigolo) da tre cellule preesistenti;
- una cellula muore se è circondata da meno di due cellule (morte per isolamento) o da più di tre cellule (morte per sovraffollamento);
- una cellula sopravvive se è circondata da due o tre cellule.

Esaminati tutti i quadretti del foglio si passa a cancellare le cellule morte e a disegnare le nuove nate.
Il gioco continua, di generazione in generazione, ripetendo le stesse operazioni, finché tutte le cellule muoiono o si arriva ad una configurazione stabile o ciclica.
Nel nostro caso il foglio quadrettato e costituito dai mille caratteri dello schermo del PET (il programma gira con soli 8 Kbyte di RAM ed è facilmente adattabile, come vedremo, al PET vecchie ROM).
Chiunque voglia cimentarsi con la stesura di un programmino di questo tipo interamente in BASIC noterà che la scrittura del programma è estremamente facile, ma la velocità di elaborazione è troppo bassa e tale da obbligare a ridurre drasticamente il numero di quadretti disponibili per portarsi a livelli accettabili.
Volendo invece mantenere l'estensione a 1.000 quadretti è necessario affidare il grosso dell'elaborazione a subroutine in linguaggio macchina. In questo modo la situazione cambia drasticamente: si giunge ad una simulazione "in tempo reale", ottenendo (in questo caso) ben due generazioni al secondo, e il gioco viene ad assumere, oltre ai significati già esposti, anche un divertente aspetto "psichedelico" nel susseguirsi rapidissimo di generazioni spesso diversissime fra loro.
Alla parte in BASIC sono affidate le routine di prompting (conversazione con l'operatore), la routine che consente l'impostazione sullo schermo, con i normali controlli del cursore, della configurazione iniziale, e le routine di controllo che consentono di sospendere l'elaborazione tenendo premuto il tasto di shift, e di interromperla premendo il tasto S.
Alle tre subroutine in linguaggio macchina è affidato il compito rispettivamente di:

- trascrivere (all'inizio) la RAM video in una zona di RAM che chiamiamo RAM AUX;
- calcolare la configurazione successiva esplorando tutti i quadretti dello schermo (tranne quelli del bordo ed eventuali "isole" di mondo esterno volute dall'utilizzatore);
- trascrivere nella RAM VIDEO il contenuto della RAM AUX, elaborato dalla subroutine precedente.

Per poter allocare queste tre subroutine e la RAM AUX si devono inserire nel programma delle istruzioni che abbassino il tetto della RAM accessibile al BASIC a soli 6655 byte (sufficienti per contenere la parte BASIC del programma), e che ripristino al termine dell'esecuzione i puntatori della massima RAM disponibile ai valori originari.
Leggendo il listing, il lettore potrà scoprire parecchie *PEEK* e *POKE* insolite. Per comodità vengono elencate delle locazioni di memoria usate dal sistema.operativo che vengono testate o modificate nel corso del programma; figura 1, fra parentesi la corrispondente locazione per il PET vecchie ROM.
Per finire, il programma può essere facilmente adattato al PET con le vecchie ROM: sarà sufficiente operare i seguenti cambiamenti:
— sostituire negli argomenti delle *PEEK* e delle *POKE*, ai valori presenti, quelli indicati fra parentesi nell'elenco precedente;
— invertire maiuscole e minuscole nelle righe di stampa.

48 (130) **49 (131)**	*Puntatore (byte basso-byte alto) all'inizio delle stringhe.*
50 (132) **51 (133)**	*Puntatore (byte basso-byte alto) alla fine delle stringhe.*
52 (134) **53 (135)**	*Valore max. (byte basso-byte alto) della RAM disponibile.*
152 (516)	*Vale 1 se è premuto il tasto di shift; altrimenti 0.*
167 (551)	*Contiene 0 se il cursore è attivato, altrimenti contiene 1; è possibile modificarlo a piacere, attivando il cursore durante l'elaborazione.*
158 (525)	*Contiene il numero di caratteri (max. 10) presenti nel buffer di tastiera: mettendoci 0 tutti i caratteri presenti vengono "eliminati".*
196 (224) **197 (225)**	*Puntano nella RAM video al primo carattere della riga (di 40 od 80 caratteri) in cui si trova il cursore.*
198 (226)	*Equivale alla POS(0): indica la posizione del cursore nella riga in corso (da 0 a 79).*
da 32768 **a 33767**	*VIDEO RAM (si ricordi che nella Video RAM i caratteri non sono codificati col codice ASCII usuale, ma secondo un codice leggermente diverso, in cui il 7° bit rappresenta i caratteri shiftati e l'8° bit rappresenta il reverse).*
59468	*Questo byte del VIA (chip 6522, che principalmente gestisce la user port) nei bit 3, 2, 1 (i primi due, 3 e 2 contengono sempre 1) comanda il generatore di caratteri, decidendo se si vuole il set grafica-maiuscole od il set maiuscole-minuscole. In pratica per avere grafica-maiuscole: POKE 59468,12 (bit 1 cont. 0) maiuscole-minuscole: POKE 59468,14 (bit 1 cont. 1).*

Figura 1 - Elenco delle locazioni di memoria utilizzate dal sistema operativo.

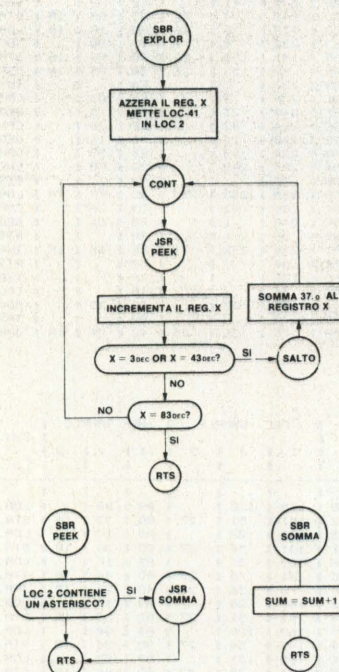
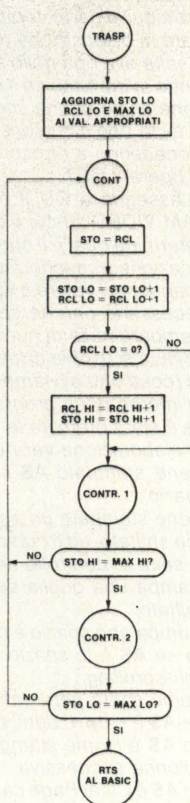
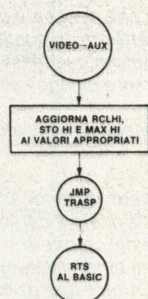
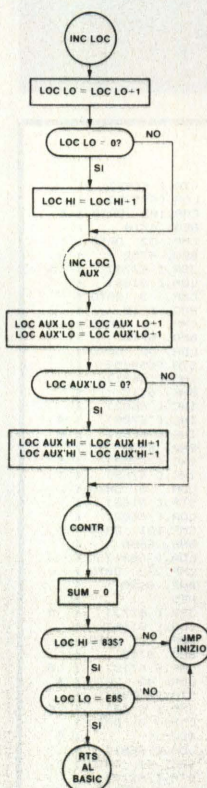
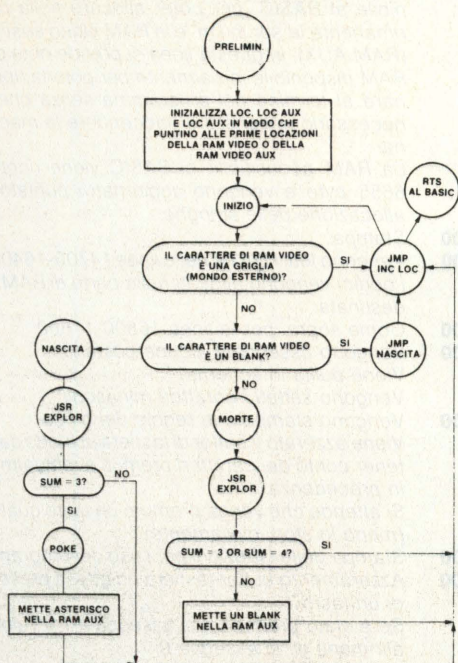
PRELIMIN.
INIZIALIZZA LOC, LOC AUX E LOC AUX IN MODO CHE PUNTINO ALLE PRIME LOCAZIONI DELLA RAM VIDEO O DELLA RAM VIDEO AUX
INIZIO
RTS AL BASIC
IL CARATTERE DI RAM VIDEO È UNA GRIGLIA (MONDO ESTERNO)?
SI
JMP INC LOC
NO
IL CARATTERE DI RAM VIDEO È UN BLANK?
SI
JMP NASCITA
NASCITA
NO
JSR EXPLOR
MORTE
SUM = 3?
NO
JSR EXPLOR
SI
POKE
SUM = 3 OR SUM = 4?
SI
NO
METTE ASTERISCO NELLA RAM AUX
METTE UN BLANK NELLA RAM AUX

VIDEO→AUX
AGGIORNA RCLHI, STO HI E MAX HI AI VALORI APPROPRIATI
JMP TRASP
RTS AL BASIC

TRASP
AGGIORNA STO LO RCL LO E MAX LO AI VAL. APPROPRIATI
CONT
STO = RCL
STO LO = STO LO+1
RCL LO = RCL LO+1
RCL LO = 0?
NO
SI
RCL HI = RCL HI+1
STO HI = STO HI+1
CONTR. 1
NO
STO HI = MAX HI?
SI
CONTR. 2
NO
STO LO = MAX LO?
SI
RTS AL BASIC

AUX → VIDEO
AGGIORNA RCL HI, STO HI e MAX HI AI VAL. APPROPRIATI
JMP TRASP
RTS AL BASIC

Figura 2 - *I flow-chart relativi al programma.*

Un consiglio: una volta che avrete avuto la certosina pazienza di trascrivere tutto il programma, non lasciatevi tentare dalla curiosità di vederlo subito girare, ma registratelo prima su nastro o su disco. Infatti basta che abbiate sbagliato a trascrivere anche uno solo dei codici delle subroutine in linguaggio macchina, contenuti nelle frasi *DATA*, perché il computer "impazzisca" e non ritorni più al BASIC, costringendovi a spegnerlo e perdendo così tutto il lavoro fatto.

Address DEC.	Address HEX.	DEC. SAMPLE 1	DEC. SAMPLE 2	DEC. SAMPLE 3	HEX SAMPLE 1	HEX SAMPLE 2	HEX SAMPLE 3	SYM,ADDR.MODE	ASCII
6656	1A00	169	28		A9	1C		LDA, 28 DATO	
6658	1A02	141	53	26	8D	35	1A	STA,[6709]	5
6661	1A05	141	113	26	8D	71	1A	STA,[6769]	1
6664	1A08	169	0		A9	00		LDA, 0 DATO	
6666	1A0A	141	52	26	8D	34	1A	STA,[6708]	4
6669	1A0D	141	112	26	8D	70	1A	STA,[6768]	0
6672	1A10	141	25	26	8D	19	1A	STA,[6681]	
6675	1A13	169	128		A9	80		LDA,128 DATO	

Life: la vita continua

```
6677 ¦ 1A15 ¦ 141 ¦  26 ¦  26 ¦ 8D ¦ 1A ¦ 1A ¦ STA,[ 6682]   ¦
6680 ¦ 1A18 ¦ 173 ¦ 255 ¦ 255 ¦ AD ¦ FF ¦ FF ¦ LDA,[65535]   ¦
6683 ¦ 1A1B ¦ 201 ¦ 102 ¦     ¦ C9 ¦ 66 ¦    ¦ CMP,102  DATO ¦ &
6685 ¦ 1A1D ¦ 240 ¦  23 ¦     ¦ F0 ¦ 17 ¦    ¦ BEQ, 6710     ¦
6687 ¦ 1A1F ¦ 201 ¦  32 ¦     ¦ C9 ¦ 20 ¦    ¦ CMP, 32  DATO ¦
6689 ¦ 1A21 ¦ 240 ¦  61 ¦     ¦ F0 ¦ 3D ¦    ¦ BEQ, 6752     ¦ =
6691 ¦ 1A23 ¦  32 ¦ 117 ¦  26 ¦ 20 ¦ 75 ¦ 1A ¦ JSR,[ 6773]   ¦  5
6694 ¦ 1A26 ¦ 173 ¦ 232 ¦  31 ¦ AD ¦ E8 ¦ 1F ¦ LDA,[ 8168]   ¦
6697 ¦ 1A29 ¦ 201 ¦   3 ¦     ¦ C9 ¦ 03 ¦    ¦ CMP,  3  DATO ¦
6699 ¦ 1A2B ¦ 240 ¦   9 ¦     ¦ F0 ¦ 09 ¦    ¦ BEQ, 6710     ¦
6701 ¦ 1A2D ¦ 201 ¦   4 ¦     ¦ C9 ¦ 04 ¦    ¦ CMP,  4  DATO ¦
6703 ¦ 1A2F ¦ 240 ¦   5 ¦     ¦ F0 ¦ 05 ¦    ¦ BEQ, 6710     ¦
6705 ¦ 1A31 ¦ 169 ¦  32 ¦     ¦ A9 ¦ 20 ¦    ¦ LDA, 32  DATO ¦
6707 ¦ 1A33 ¦ 141 ¦ 255 ¦ 255 ¦ 8D ¦ FF ¦ FF ¦ STA,[65535]   ¦
6710 ¦ 1A36 ¦ 238 ¦  25 ¦  26 ¦ EE ¦ 19 ¦ 1A ¦ INC,[ 6681]   ¦
6713 ¦ 1A39 ¦ 208 ¦   3 ¦     ¦ D0 ¦ 03 ¦    ¦ BNE, 6718     ¦
6715 ¦ 1A3B ¦ 238 ¦  26 ¦  26 ¦ EE ¦ 1A ¦ 1A ¦ INC,[ 6682]   ¦
6718 ¦ 1A3E ¦ 238 ¦  52 ¦  26 ¦ EE ¦ 34 ¦ 1A ¦ INC,[ 6708]   ¦ 4
6721 ¦ 1A41 ¦ 238 ¦ 112 ¦  26 ¦ EE ¦ 70 ¦ 1A ¦ INC,[ 6768]   ¦ 0
6724 ¦ 1A44 ¦ 208 ¦   6 ¦     ¦ D0 ¦ 06 ¦    ¦ BNE, 6732     ¦
6726 ¦ 1A46 ¦ 238 ¦  53 ¦  26 ¦ EE ¦ 35 ¦ 1A ¦ INC,[ 6709]   ¦ 5
6729 ¦ 1A49 ¦ 238 ¦ 113 ¦  26 ¦ EE ¦ 71 ¦ 1A ¦ INC,[ 6769]   ¦ 1
6732 ¦ 1A4C ¦ 169 ¦   0 ¦     ¦ A9 ¦ 00 ¦    ¦ LDA,  0  DATO ¦
6734 ¦ 1A4E ¦ 141 ¦ 232 ¦  31 ¦ 8D ¦ E8 ¦ 1F ¦ STA,[ 8168]   ¦
6737 ¦ 1A51 ¦ 173 ¦  26 ¦  26 ¦ AD ¦ 1A ¦ 1A ¦ LDA,[ 6682]   ¦
6740 ¦ 1A54 ¦ 201 ¦ 131 ¦     ¦ C9 ¦ 83 ¦    ¦ CMP,131  DATO ¦
6742 ¦ 1A56 ¦ 208 ¦ 192 ¦     ¦ D0 ¦ C0 ¦    ¦ BNE, 6680     ¦
6744 ¦ 1A58 ¦ 173 ¦  25 ¦  26 ¦ AD ¦ 19 ¦ 1A ¦ LDA,[ 6681]   ¦
6747 ¦ 1A5B ¦ 201 ¦ 232 ¦     ¦ C9 ¦ E8 ¦    ¦ CMP,232  DATO ¦
6749 ¦ 1A5D ¦ 208 ¦ 185 ¦     ¦ D0 ¦ B9 ¦    ¦ BNE, 6680     ¦
6751 ¦ 1A5F ¦  96 ¦     ¦     ¦ 60 ¦    ¦    ¦ RTS           ¦
6752 ¦ 1A60 ¦  32 ¦ 117 ¦  26 ¦ 20 ¦ 75 ¦ 1A ¦ JSR,[ 6773]   ¦  5
6755 ¦ 1A63 ¦ 173 ¦ 232 ¦  31 ¦ AD ¦ E8 ¦ 1F ¦ LDA,[ 8168]   ¦
6758 ¦ 1A66 ¦ 201 ¦   3 ¦     ¦ C9 ¦ 03 ¦    ¦ CMP,  3  DATO ¦
6760 ¦ 1A68 ¦ 240 ¦   3 ¦     ¦ F0 ¦ 03 ¦    ¦ BEQ, 6765     ¦
6762 ¦ 1A6A ¦  76 ¦  54 ¦  26 ¦ 4C ¦ 36 ¦ 1A ¦ JMP,[ 6710]   ¦ L6
6765 ¦ 1A6D ¦ 169 ¦  42 ¦     ¦ A9 ¦ 2A ¦    ¦ LDA, 42  DATO ¦ *
6767 ¦ 1A6F ¦ 141 ¦ 255 ¦ 255 ¦ 8D ¦ FF ¦ FF ¦ STA,[65535]   ¦
6770 ¦ 1A72 ¦  76 ¦  54 ¦  26 ¦ 4C ¦ 36 ¦ 1A ¦ JMP,[ 6710]   ¦ L6
6773 ¦ 1A75 ¦ 162 ¦   0 ¦     ¦ A2 ¦ 00 ¦    ¦ LDX,  0  DATO ¦
6775 ¦ 1A77 ¦  56 ¦     ¦     ¦ 38 ¦    ¦    ¦ SEC           ¦ 8
6776 ¦ 1A78 ¦ 173 ¦  25 ¦  26 ¦ AD ¦ 19 ¦ 1A ¦ LDA,[ 6681]   ¦
6779 ¦ 1A7B ¦ 233 ¦  41 ¦     ¦ E9 ¦ 29 ¦    ¦ SBC, 41  DATO ¦ )
6781 ¦ 1A7D ¦ 141 ¦ 154 ¦  26 ¦ 8D ¦ 9A ¦ 1A ¦ STA,[ 6810]   ¦
6784 ¦ 1A80 ¦ 173 ¦  26 ¦  26 ¦ AD ¦ 1A ¦ 1A ¦ LDA,[ 6682]   ¦
6787 ¦ 1A83 ¦ 233 ¦   0 ¦     ¦ E9 ¦ 00 ¦    ¦ SBC,  0  DATO ¦
6789 ¦ 1A85 ¦ 141 ¦ 155 ¦  26 ¦ 8D ¦ 9B ¦ 1A ¦ STA,[ 6811]   ¦
6792 ¦ 1A88 ¦  32 ¦ 153 ¦  26 ¦ 20 ¦ 99 ¦ 1A ¦ JSR,[ 6809]   ¦
6795 ¦ 1A8B ¦ 232 ¦     ¦     ¦ E8 ¦    ¦    ¦ INX           ¦
6796 ¦ 1A8C ¦ 224 ¦   3 ¦     ¦ E0 ¦ 03 ¦    ¦ CPX,  3  DATO ¦
6798 ¦ 1A8E ¦ 240 ¦  21 ¦     ¦ F0 ¦ 15 ¦    ¦ BEQ, 6821     ¦
6800 ¦ 1A90 ¦ 224 ¦  43 ¦     ¦ E0 ¦ 2B ¦    ¦ CPX, 43  DATO ¦ +
6802 ¦ 1A92 ¦ 240 ¦  17 ¦     ¦ F0 ¦ 11 ¦    ¦ BEQ, 6821     ¦
6804 ¦ 1A94 ¦ 224 ¦  83 ¦     ¦ E0 ¦ 53 ¦    ¦ CPX, 83  DATO ¦ S
6806 ¦ 1A96 ¦ 208 ¦ 240 ¦     ¦ D0 ¦ F0 ¦    ¦ BNE, 6792     ¦
6808 ¦ 1A98 ¦  96 ¦     ¦     ¦ 60 ¦    ¦    ¦ RTS           ¦
6809 ¦ 1A99 ¦ 189 ¦ 255 ¦ 255 ¦ BD ¦ FF ¦ FF ¦ LDA,[65535+X] ¦
6812 ¦ 1A9C ¦ 201 ¦  42 ¦     ¦ C9 ¦ 2A ¦    ¦ CMP, 42  DATO ¦ *
6814 ¦ 1A9E ¦ 240 ¦   1 ¦     ¦ F0 ¦ 01 ¦    ¦ BEQ, 6817     ¦
6816 ¦ 1AA0 ¦  96 ¦     ¦     ¦ 60 ¦    ¦    ¦ RTS           ¦
6817 ¦ 1AA1 ¦ 238 ¦ 232 ¦  31 ¦ EE ¦ E8 ¦ 1F ¦ INC,[ 8168]   ¦
6820 ¦ 1AA4 ¦  96 ¦     ¦     ¦ 60 ¦    ¦    ¦ RTS           ¦
6821 ¦ 1AA5 ¦ 138 ¦     ¦     ¦ 8A ¦    ¦    ¦ TXA           ¦
6822 ¦ 1AA6 ¦  24 ¦     ¦     ¦ 18 ¦    ¦    ¦ CLC           ¦
6823 ¦ 1AA7 ¦ 105 ¦  37 ¦     ¦ 69 ¦ 25 ¦    ¦ ADC, 37  DATO ¦ )%
6825 ¦ 1AA9 ¦ 170 ¦     ¦     ¦ AA ¦    ¦    ¦ TAX           ¦
6826 ¦ 1AAA ¦  76 ¦ 136 ¦  26 ¦ 4C ¦ 88 ¦ 1A ¦ JMP,[ 6792]   ¦ L
```

```
  Address     ¦    DEC. SAMPLE    ¦  HEX SAMPLE  ¦                 ¦
              ¦                   ¦              ¦ SYM.ADDR.MODE   ¦ ASCII
 DEC. ¦ HEX.  ¦  1  ¦  2  ¦  3    ¦ 1 ¦  2 ¦  3  ¦                 ¦
      ¦       ¦     ¦     ¦       ¦   ¦    ¦     ¦                 ¦
-----------------------------------------------------------------------------
      ¦       ¦     ¦     ¦       ¦    ¦    ¦    ¦                 ¦
6912 ¦ 1B00 ¦ 169 ¦ 128 ¦     ¦ A9 ¦ 80 ¦    ¦ LDA,128  DATO ¦
6914 ¦ 1B02 ¦ 141 ¦  51 ¦  27 ¦ 8D ¦ 33 ¦ 1B ¦ STA,[ 6963]   ¦ 3
6917 ¦ 1B05 ¦ 169 ¦  28 ¦     ¦ A9 ¦ 1C ¦    ¦ LDA, 28  DATO ¦
6919 ¦ 1B07 ¦ 141 ¦  54 ¦  27 ¦ 8D ¦ 36 ¦ 1B ¦ STA,[ 6966]   ¦ 6
6922 ¦ 1B0A ¦ 169 ¦  31 ¦     ¦ A9 ¦ 1F ¦    ¦ LDA, 31  DATO ¦
6924 ¦ 1B0C ¦ 141 ¦  70 ¦  27 ¦ 8D ¦ 46 ¦ 1B ¦ STA,[ 6982]   ¦ F
6927 ¦ 1B0F ¦  76 ¦  36 ¦  27 ¦ 4C ¦ 24 ¦ 1B ¦ JMP,[ 6948]   ¦ L$
6930 ¦ 1B12 ¦ 169 ¦  28 ¦     ¦ A9 ¦ 1C ¦    ¦ LDA, 28  DATO ¦
6932 ¦ 1B14 ¦ 141 ¦  51 ¦  27 ¦ 8D ¦ 33 ¦ 1B ¦ STA,[ 6963]   ¦ 3
6935 ¦ 1B17 ¦ 169 ¦ 128 ¦     ¦ A9 ¦ 80 ¦    ¦ LDA,128  DATO ¦
6937 ¦ 1B19 ¦ 141 ¦  54 ¦  27 ¦ 8D ¦ 36 ¦ 1B ¦ STA,[ 6966]   ¦ 6
6940 ¦ 1B1C ¦ 169 ¦ 131 ¦     ¦ A9 ¦ 83 ¦    ¦ LDA,131  DATO ¦
6942 ¦ 1B1E ¦ 141 ¦  70 ¦  27 ¦ 8D ¦ 46 ¦ 1B ¦ STA,[ 6982]   ¦ F
6945 ¦ 1B21 ¦  76 ¦  36 ¦  27 ¦ 4C ¦ 24 ¦ 1B ¦ JMP,[ 6948]   ¦ L$
6948 ¦ 1B24 ¦ 169 ¦   0 ¦     ¦ A9 ¦ 00 ¦    ¦ LDA,  0  DATO ¦
6950 ¦ 1B26 ¦ 141 ¦  53 ¦  27 ¦ 8D ¦ 35 ¦ 1B ¦ STA,[ 6965]   ¦ 5
6953 ¦ 1B29 ¦ 141 ¦  50 ¦  27 ¦ 8D ¦ 32 ¦ 1B ¦ STA,[ 6962]   ¦ 2
6956 ¦ 1B2C ¦ 169 ¦ 232 ¦     ¦ A9 ¦ E8 ¦    ¦ LDA,232  DATO ¦
6958 ¦ 1B2E ¦ 141 ¦  80 ¦  27 ¦ 8D ¦ 50 ¦ 1B ¦ STA,[ 6992]   ¦ P
6961 ¦ 1B31 ¦ 173 ¦   0 ¦   0 ¦ AD ¦ 00 ¦ 00 ¦ LDA,[    0]   ¦
6964 ¦ 1B34 ¦ 141 ¦   0 ¦   0 ¦ 8D ¦ 00 ¦ 00 ¦ STA,[    0]   ¦
6967 ¦ 1B37 ¦ 238 ¦  53 ¦  27 ¦ EE ¦ 35 ¦ 1B ¦ INC,[ 6965]   ¦ 5
6970 ¦ 1B3A ¦ 238 ¦  50 ¦  27 ¦ EE ¦ 32 ¦ 1B ¦ INC,[ 6962]   ¦ 2
6973 ¦ 1B3D ¦ 208 ¦   6 ¦     ¦ D0 ¦ 06 ¦    ¦ BNE, 6981     ¦
6975 ¦ 1B3F ¦ 238 ¦  54 ¦  27 ¦ EE ¦ 36 ¦ 1B ¦ INC,[ 6966]   ¦ 6
6978 ¦ 1B42 ¦ 238 ¦  51 ¦  27 ¦ EE ¦ 33 ¦ 1B ¦ INC,[ 6963]   ¦ 3
6981 ¦ 1B45 ¦ 169 ¦   0 ¦     ¦ A9 ¦ 00 ¦    ¦ LDA,  0  DATO ¦
6983 ¦ 1B47 ¦ 205 ¦  54 ¦  27 ¦ CD ¦ 36 ¦ 1B ¦ CMP,[ 6966]   ¦ 6
6986 ¦ 1B4A ¦ 240 ¦   3 ¦     ¦ F0 ¦ 03 ¦    ¦ BEQ, 6991     ¦
6988 ¦ 1B4C ¦  76 ¦  49 ¦  27 ¦ 4C ¦ 31 ¦ 1B ¦ JMP,[ 6961]   ¦ L1
6991 ¦ 1B4F ¦ 169 ¦   0 ¦     ¦ A9 ¦ 00 ¦    ¦ LDA,  0  DATO ¦
6993 ¦ 1B51 ¦ 205 ¦  53 ¦  27 ¦ CD ¦ 35 ¦ 1B ¦ CMP,[ 6965]   ¦ 5
6996 ¦ 1B54 ¦ 240 ¦   3 ¦     ¦ F0 ¦ 03 ¦    ¦ BEQ, 7001     ¦
6998 ¦ 1B56 ¦  76 ¦  49 ¦  27 ¦ 4C ¦ 31 ¦ 1B ¦ JMP,[ 6961]   ¦ L1
7001 ¦ 1B59 ¦  96 ¦     ¦     ¦ 60 ¦    ¦    ¦ RTS           ¦
```

Listato in linguaggio macchina dal programma life.

REMark	
1000-2300	*REM iniziali.*
2400	*Poiché il pgm abbasserà il tetto della RAM disponibile al BASIC, per poter allocare nella parte rimanente le sbr in l.m. e la RAM video ausiliaria (RAM AUX), in questa linea si prende nota della RAM disponibile inizialmente per poterla ripristinare al termine del programma senza che sia necessario spegnere e riaccendere la macchina.*
2500	*La RAM accessibile al BASIC viene ridotta a 6655 byte e vengono aggiornati i puntatori di allocazione delle stringhe.*
2600-2700	*Stampa.*
2800-3000	*Vengono letti i DATA delle linee 14700-16400 ed i codici vengono allocati nella parte di RAM loro destinata.*
3100-3300	*Come sopra, per le linee 16800-17600.*
3400-3500	*Vengono assegnate alcune costanti.*
3600	*Viene pulito lo schermo.*
3700	*Vengono settati i caratteri minuscoli.*
3800-5700	*Vengono stampate le regole del gioco.*
5800	*Viene azzerato il buffer di tastiera in modo da non tener conto dei caratteri premuti eventualmente in precedenza.*
5900	*Si attende che venga premuto un tasto qualsiasi (meno lo stop, ovviamente).*
6000-8300	*Stampa delle istruzioni per l'uso del programma.*
8400-8500	*Azzeramento buffer tastiera ed attesa pressione di un tasto.*
8600	*Se è stato premuto n si assegna a T il valore 1, altrimenti gli si assegna 0.*
8700	*Si cancella la pagina e si predispone la posizione del cursore.*
8800	*Si settano i caratteri grafici.*
8900	*Si esegue la sbr che disegna la cornice ai bordi dello schermo.*
9000	*SI provoca l'azionamento del cursore lampeggiante.*
9100-9300	*Si pongono direttamente nella RAM VIDEO i pallini di riferimento che indicano i punti di mezzo della cornice.*
9400	*Inizia qui la parte iterativa che consente di impostare la disposizione di partenza. In questa linea si salta alla riga dello schermo successiva (evitando la cornice) se il cursore giunge sulla locazione finale di una linea.*
9500	*Si evita che il cursore possa finire sulla cornice procedendo a ritroso.*
9600	*Si opera il GET.*
9700	*Si assegna a CU il numero della locazione in RAM VIDEO su cui è posizionato il cursore, e si determina qual'è il codice del carattere in questa locazione (mascherando il bit di reverse), assegnandone il valore a VC. Queste operazioni sono necessarie perché spostando il cursore senza stampare nulla di nuovo nella locazione in cui si trovava si rischia di lasciare il quadretto in reverse (cosa che ovviamente accade solo operando all'interno di un programma).* *Se è stata premuta la freccia a sinistra si passa all'elaborazione vera e propria.*
9800	*Viene stampato A$ se è un asterisco od uno spazio.*
9900	*Viene stampato un asterisco se A$ è un asterisco shiftato, ed il cursore è posizionato in basso. Si salta al controllo anti-scrolling.*
10000	*Stampa una griglia se A$ è il tasto griglia non shiftato.*
10100	*Stampa uno spazio e posiziona il cursore in basso se A$ è lo spazio shiftato. Salta al controllo anti-scrolling.*
10200	*Come sopra per il tasto griglia shiftato.*
10300	*Se A$ è c.up, c.right, c.left salta alla riga 10800.*
10400	*Se A$ è Home stampa A$ e si predispone alla colonna successiva.*
10500	*Se A$ è Clear Page cancella la pagina e rigenera la cornice.*
10600	*Permette di stampare un c.down od un carriage return (shiftato o meno) solo se facendo ciò non si finisce sulla cornice.*
10700	*Nel caso si tratti di un carattere non ammesso non lo considera e torna ad accettare altri caratteri.*

10800	*Stampa dei caratteri di movimento del cursore con il controllo del reverse sulla casella di partenza.*
10900-11000	*Durante il modo di scrittura verticale provoca il "carriage return" alla colonna successiva per evitare lo scrolling.*
11100	*Disattiva il cursore ripulendo l'eventuale reverse.*
11200	*Rigenera la cornice (indispensabile per il corretto funzionamento delle sbr in linguaggio macchina).*
11300	*Trascrive la RAM video nella RAM AUX.*
11400	*Punto di inizio del processo ricorsivo di generazione; viene calcolata la nuova configurazione lasciandola nella RAM AUX.*
11500	*La variabile PE assume il valore del flag di shift.*
11600	*In caso di necessità (richiesta iniziale o pressione del tasto di shift) si salta alla sbr che stampa il n. della gen.*
11700	*Se il tasto di shift non è premuto salta la parte destinata ad attendere il suo rilascio.*
11800-12000	*Loop di attesa interrotto dal rilascio del tasto di shift o dalla pressione di "S". La struttura un po' strana di questa parte del pgm è legata al fatto che la pressione di "S" con shift premuto, a causa della routine di scansione della tastiera, provoca interferenze sul flag di shift.*
12100	*Viene testato "S" per il caso in cui non sia premuto shift.*
12200	*Ad ogni generazione viene azzerato il buffer di tastiera.*
12300	*Viene trascritta la RAM AUX nella RAM VIDEO, visualizzando così la generazione calcolata alla linea 11400.*
12400	*Viene aggiornato il numero della generazione.*
12500	*Si passa all'elaborazione della generazione successiva.*
12600-13400	*Viene disegnata la cornice, servendosi della sbr 14300 e assegnando i valori di I1 (loc. iniziale), I2 (loc. finale) e I3 (distanza fra le griglie).*
13500-14200	*Dopo la pressione di "S" si domanda se si vogliono aggiungere altre cellule (salto alla riga 9000) o se si vuole uscire dal programma (salto alla riga 16500)*
14300-14600	*Stampa una serie di griglie alle locazioni indicate dai parametri I1 / I2 e I3.*
14700-16400	*Codici della sbr in l.m. che si occupa del calcolo della generazione successiva.*
16500-16700	*Vengono resettati i puntatori di max RAM e di allocazione delle stringhe, e si esce dal programma.*
16800-17600	*Codici delle sbr. di trascrizione fra RAM VIDEO e RAM AUX.*
17700-18100	*Viene scritto il n. della generazione in corso. Si attende qualche istante e si torna. Se viene premuto shift si aggiorna PE e si torna immediatamente alla linea di partenza.*

```
 1000 REM *************
 1100 REM *           *
 1200 REM *  L I F E  *
 1300 REM *           *
 1400 REM *************
 1500 REM
 2300 REM
 2400 ML=PEEK(52):MH=PEEK(53)
 2500 POKE52,255:POKE53,25:POKE50,255:P
OKE51,25:POKE48,255:POKE49,25
 2600 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒";TAB(15);"______
____"
 2700 PRINTTAB(15);"▒* LIFE *"
 2800 FORI=6656TO6828
 2900 READDD:POKEI,DD
 3000 NEXT
 3100 FORI=6912TO7001
 3200 READDD:POKEI,DD
 3300 NEXT
 3400 A=32768:B=33767:DI=25600
 3500 S1=102
```

```
 3600 PRINT"▒"
 3700 POKE59468,14
 3800 PRINTSPC(13);"▒           "
 3900 PRINTSPC(13);"▒ * L ╮ — ‾ * "
 4000 PRINTSPC(13);"▒           "
 4100 PRINT:PRINT"●UESTO PROGRAMMA SIMU
LA LO SVILUPPO"
 4200 PRINT"DELLA ╳ITA IN UNO SPAZIO BI
DIMENSIONALE"
 4300 PRINT"COSTITUITO DAI  1000  CARAT
TERI  DELLO"
 4400 PRINT"SCHERMO  DEL ′┬┤′."
 4500 PRINT"┐ATA UNA CONFIGURAZIONE INI
ZIALE SCELTA"
 4600 PRINT"DALL′ ╱TENTE, LE CELLULE SI
SVILUPPANO"
 4700 PRINT"SEGUENDO QUESTE REGOLE:"
 4800 PRINT:PRINT"-  ╱NA CELLULA NASCE I
N OGNI SPAZIO VUO-"
 4900 PRINT"  TO CIRCONDATO (DI LATO O
DI SPIGOLO)"
 5000 PRINT"  DA TRE CELLULE PREESISTEN
TI;"
 5100 PRINT"-  ╱NA CELLULE MUORE SE E′ C
IRCONDATA DA"
 5200 PRINT"  MENO DI DUE O PIU′ DI TRE
 CELLULE;"
 5300 PRINT"-  ╱NA CELLULA SOPRAVVIVE SE
 E′ CIRCON-"
 5400 PRINT"  DATA DA DUE O TRE ALTRE C
ELLULE;"
 5500 PRINT"- ╱ESSUNA CELLULA PUO′ VIVE
RE NEGLI"
 5600 PRINT"  SPAZI COSTITUENTI IL ′╲ON
DO ‾STERNO′."
 5700 PRINT:PRINT"┐REMERE UN TASTO PER
CONTINUARE.";
 5800 POKE158,0
 5900 GETA$:IFA$=""THEN5900
 6000 PRINT"▒";
 6100 PRINT"┐OPO CHE AVRETE LETTO QUEST
A PAGINA"
 6200 PRINT"IL ′┬┤′ SI PREDISPORRA′ AD
 ACCETTARE"
 6300 PRINT"IL DISEGNO DELLA CONFIGURAZ
IONE INIZIA-"
 6400 PRINT"LE, CHE SARA′ EFFETTUATO SE
MPLICEMENTE"
 6500 PRINT"PONENDO IL CARATTERE * NEI
QUADRETTI"
 6600 PRINT"CHE SI VUOLE SIANO OCCUPATI
 DA CELLULE"
 6700 PRINT"ED IL CARATTERE ▒ IN QUELLI
CHE DEBBONO"
 6800 PRINT"RAPPRESENTARE EVENTUALI ′IS
OLE′ DI MON-"
 6900 PRINT"DO  ESTERNO  DENTRO  LO  SC
HERMO."
 7000 PRINT"♥I POSSONO  USARE, OLTRE  A
I TASTI DEI"
 7100 PRINT"CARATTERI SUDDETTI, CHE SHI
FTATI PRODU-"
 7200 PRINT"CONO COLONNE VERTICALI, ANC
HE I NORMALI"
 7300 PRINT"CONTROLLI DEL CURSORE, LO ♥
┐♣♦╮┌ (PER"
 7400 PRINT"CANCELLARE), IL TASTO _┐│ ╱—
╱ ED IL TA-"
 7500 PRINT"STO ┤└_- │┌╲‾. ┐REMENDO ← SI
```

```
 DA' INIZIO"
 7600 PRINT"ALL'ELABORAZIONE. TREMENDO
'♥ ⊢-|'"
 7700 PRINT"SI OTTIENE UNA SOSPENSIONE
TEMPORANEA,"
 7800 PRINT"MENTRE PREMENDO '♥' L'ELA
BORAZIONE SI"
 7900 PRINT"ARRESTA, CONSENTENDO DI
MODIFICARE"
 8000 PRINT"LA SITUAZIONE O DI USCIRE D
AL PROGRAMMA."
 8100 PRINT"♥E SI VUOLE VISUALIZZARE IL
NUMERO DI"
 8200 PRINT"OGNI GENERAZIONE PREMERE 'N
';"
 8300 PRINT"ALTRIMENTI PREMERE UN ALTRO
TASTO."
 8400 POKE158,0
 8500 GETA$:IFA$=""THEN8500
 8600 T=0:IFA$="N"THENT=1
 8700 PRINT"■■■";
 8800 POKE59468,12
 8900 GOSUB12600
 9000 POKE167,0
 9100 POKE33248,209:POKE33287,209
 9200 POKE33747,209:POKE33748,209
 9300 POKE32787,209:POKE32788,209
 9400 IFPEEK(198)=39ORPEEK(198)=79THENP
RINT"■■";
 9500 IFPEEK(198)=0ORPEEK(198)=40THENPR
INT"■";
 9600 GETA$:IFA$=""THEN9400
 9700 CU=256*PEEK(197)+PEEK(196)+PEEK(1
98):VC=PEEK(CU)AND127:IFA$="←"THEN11100
 9800 IFA$="*"ORA$=" "THENPRINTA$;:GOTO
9400
 9900 IFA$=" |"THENPRINT"*■■";:GOTO10900
 10000 IFA$="&"THENPRINT"▒";:GOTO9400
 10100 IFASC(A$)=160THENPRINT" ■■";:GOT
O10900
 10200 IFA$="▒"THENPRINT"▒■■";:GOTO1090
0
 10300 IFA$="□"ORA$="■"ORA$="■"THEN1080
0
 10400 IFA$="■"THENPRINTA$:POKECU,VC:GO
TO9400
 10500 IFA$="□"THENPRINTA$:GOSUB12600:G
OTO9000
 10600 IF(A$="■"ORASC(A$)=13ORASC(A$)=1
41)ANDCU<33687THEN10800
 10700 GOTO9400
 10800 PRINTA$;:POKECU,VC:GOTO9400
 10900 IFCU>33686THENPO=POS(0):PRINT"■■
";TAB(PO+1);
 11000 GOTO9400
 11100 POKE167,1:POKECU,VC
 11200 GOSUB12600
 11300 SYS(6912)
 11400 SYS(6656)
 11500 PE=PEEK(152)
 11600 IF(T=0ANDPE=1)ORT<>0THENGOSUB177
00
 11700 IFPE=0THEN12100
 11800 GETA$
 11900 IFA$="♥"ORA$="S"THEN13500
 12000 IFPEEK(152)=1THEN11800
 12100 GETA$:IFA$="S"ORA$="♥"THEN13500
 12200 POKE158,0
 12300 SYS(6930)
 12400 G=G+1
 12500 GOTO11400
 12600 I1=A:I2=32807:I3=1
 12700 GOSUB14300
 12800 I1=33728:I2=33767
 12900 GOSUB14300
 13000 I2=I1-I3:I1=32808:I3=40
 13100 GOSUB14300
 13200 I1=32847
 13300 GOSUB14300
 13400 RETURN
 13500 PRINT"■";
 13600 PRINTTAB(4);"GENER."G;"■";
 13700 PRINT";PER CONT.'C', INT.'S'"
 13800 POKE158,0
 13900 GETA$
 14000 IFA$="C"ORA$="-"THENPRINT"■■■";:
GOTO9000
 14100 IFA$="S"ORA$="♥"THEN16500
 14200 GOTO13900
 14300 FORI=I1TOI2STEPI3
 14400 POKEI,S1
 14500 NEXT
 14600 RETURN
 14700 DATA 169, 28, 141, 53, 26, 141,
113, 26, 169, 0, 141
 14800 DATA 52, 26, 141, 112, 26, 141,
25, 26, 169, 128
 14900 DATA 141, 26, 26, 173, 255, 255,
201, 102, 240, 23
 15000 DATA 201, 32, 240, 61, 32, 117,
26, 173, 232, 31
 15100 DATA 201, 3, 240, 9, 201, 4, 240
, 5, 169, 32
 15200 DATA 141, 255, 255, 238, 25, 26,
208, 3, 238, 26
 15300 DATA 26, 238, 52, 26, 238, 112,
26, 208, 6, 238
 15400 DATA 53, 26, 238, 113, 26, 169,
0, 141, 232, 31
 15500 DATA 173, 26, 26, 201, 131, 208,
192, 173, 25, 26
 15600 DATA 201, 232, 208, 185, 96, 32,
117, 26, 173, 232
 15700 DATA 31, 201, 3, 240, 3, 76, 54,
26, 169, 42
 15800 DATA 141, 255, 255, 76, 54, 26,
162, 0, 56, 173
 15900 DATA 25, 26, 233, 41, 141, 154,
26, 173, 26, 26
 16000 DATA 233, 0, 141, 155, 26, 32, 1
53, 26, 232, 224
 16100 DATA 3, 240, 21, 224, 43, 240, 1
7, 224, 83, 208
 16200 DATA 240, 96, 189, 255, 255, 201
, 42, 240, 1, 96
 16300 DATA 238, 232, 31, 96, 138, 24,
105, 37, 170, 76
 16400 DATA 136, 26
 16500 PRINT"■";
 16600 POKE52,ML:POKE53,MH:POKE50,ML:PO
KE51,MH:POKE48,ML:POKE49,MH
 16700 END
 16800 DATA 169, 128, 141, 51, 27, 169,
28, 141, 54, 27, 169
 16900 DATA 31, 141, 70, 27, 76, 36, 27
, 169, 28, 141
 17000 DATA 51, 27, 169, 128, 141, 54,
27, 169, 131, 141
 17100 DATA 70, 27, 76, 36, 27, 169, 0,
141, 53, 27
 17200 DATA 141, 50, 27, 169, 232, 141,
80, 27, 173, 0
 17300 DATA 0, 141, 0, 0, 238, 53, 27,
238, 50, 27
 17400 DATA 208, 6, 238, 54, 27, 238, 5
```

```
1, 27, 169, 0
 17500 DATA 205, 54, 27, 240, 3, 76, 49
, 27, 169, 0
 17600 DATA 205, 53, 27, 240, 3, 76, 49
, 27, 96
 17700 PRINT"▓";
 17800 PRINTTAB(7);"### GENERAZIONE N."
;G;"▌";
 17900 PRINT" ###"
 18000 FORI=1TO20:PE=PEEK(152):IFPE=1TH
ENRETURN
 18100 NEXT:RETURN


Lista simboli grafici

 2600  : 1   SHIFT HOME =CHR$(147)
        11   CRSR↑ =CHR$(17)

 2700  : 1   REVERSE =CHR$(18)

 3600  : 1   SHIFT HOME =CHR$(147)

 3800  : 1   REVERSE =CHR$(18)

 3900  : 1   REVERSE =CHR$(18)

 4000  : 1   REVERSE =CHR$(18)

 6000  : 1   SHIFT HOME =CHR$(147)

 8700  : 1   SHIFT HOME =CHR$(147)
         1   CRSR↑ =CHR$(17)
         1   CRSR← =CHR$(29)

 9400  : 2   CRSR← =CHR$(29)

 9500  : 1   CRSR← =CHR$(29)

 9900  : 1   CRSR↑ =CHR$(17)
             SHIFT CRSR← =CHR$(157)

10100  : 1   CRSR↑ =CHR$(17)
         1   SHIFT CRSR← =CHR$(157)

10200  : 1   CRSR↑ =CHR$(17)
         1   SHIFT CRSR← =CHR$(157)

10300  : 1   SHIFT CRSR↑ =CHR$(145)
         1   CRSR← =CHR$(29)
         1   SHIFT CRSR← =CHR$(157)

10400  : 1   HOME =CHR$(19)

10500  : 1   SHIFT HOME =CHR$(147)

10600  : 1   CRSR↑ =CHR$(17)

10900  : 1   HOME =CHR$(19)
         1   CRSR↑ =CHR$(17)

13500  : 1   HOME =CHR$(19)

13600  : 1   SHIFT CRSR← =CHR$(157)

14000  : 1   HOME =CHR$(19)
         1   CRSR↑ =CHR$(17)
         1   CRSR← =CHR$(29)

16500  : 1   HOME =CHR$(19)

17700  : 1   HOME =CHR$(19)

17800  : 1   SHIFT CRSR← =CHR$(157)
```

Listato BASIC del programma Life.

Office Automation Xerox

Xerox 860

Sistemi elettronici per l'elaborazione e la trasmissione dell'informazione.

Oggi, quando si parla di Office Automation, si pensa a una serie di apparecchiature, integrate tra loro, che permettono la elaborazione e la trasmissione dell'informazione.

E Xerox 860 è una famiglia di "stazioni di lavoro" multifunzionali per il trattamento e la trasmissione dell'informazione nella duplice forma di testi e dati. Sia nella versione a tutta pagina che in quella a mezza pagina Xerox 860 è un evoluto elaboratore di testi che consente una gestione accurata e completa della redazione e correzione di qualunque tipo di documento. Con Xerox 860 è possibile, inoltre, programmare molteplici funzioni di elaborazione di testi, gestire archivi con varie modalità di accesso e ordinamento, effettuare operazioni aritmetiche. La famiglia 860, oltre al sistema operativo Xerox, può utilizzare anche il sistema operativo CP/M®. Vengono, così, ulteriormente aumentate le sue potenzialità con l'accesso ad un'ampia gamma di pacchetti applicativi e linguaggi di programmazione forniti dalla Rank Xerox o facilmente reperibili sul mercato.

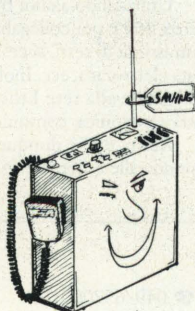
Utilizzando alcuni fra i più diffusi protocolli di comunicazione, Xerox 860 è poi collegabile con apparecchiature Xerox e non, per trasmissioni di testi, accesso a banche dati, aggiornamento di archivi, posta elettronica, ecc. Inoltre, su base locale, Xerox 860 è direttamente collegabile alla rete Ethernet e ne utilizza i servizi di archiviazione, posta elettronica, comunicazione e stampa.

Xerox 860 è dunque, la famiglia di stazioni di lavoro che meglio risponde alle odierne esigenze dell'Office Automation.

Centri Commerciali Rank Xerox. Sede: Milano ☎ 02/2883-271

Ancona	☎ 071/897661	Cernusco S/N (MI)	☎ 02/923681	Padova	☎ 049/657000
Bari	☎ 080/227499	Firenze	☎ 055/483244	Palermo	☎ 091/296241
Bologna	☎ 051/558600	Genova	☎ 010/564213	Roma	☎ 06/5461255
Cagliari	☎ 070/667708	Milano	☎ 02/2883396	Torino	☎ 011/542212
Catania	☎ 095/310263	Napoli	☎ 081/684788	Trieste	☎ 040/763841

RANK XEROX

SAVING COMPUTER '83
APPLE
DYSAN
SORD
SIRIUS
DIGITEK
COMMODORE
La sorgente
per le necessità del tuo computer
Nella nostra sala mostra
potrai ammirare e provare prodotti come:
stampanti
floppy disk
programmi
biblioteca specializzata
Le migliori marche di Personal Computer
Disponiamo infatti pronta consegna
di APPLE II, APPLE II E, SIRIUS,
SORD M 23, AVT COMP 2, VIC 20,
VIC 64, ZX81, SPECTRUM, MPF II
Non perdere
questa occasione!!!
DISTRIBUTORI ESCLUSIVI DEL FAVOLOSO
"THE LAST ONE" PER IL VENETO
Vendita anche per corrispondenza,
telefona per le quotazioni, saremo lieti di accontentarti.
SAVING
ELETTRONICA
VIA GRAMSCI 40 - MIRANO (VE) - TEL. (041) 432876

Modello di previsione delle vendite con lo ZX 80/81

Il futuro commerciale svelato da un Sinclair

di G. Bigi

```
  10 REM ***********************
  20 REM *                     *
  30 REM *  PREVISIONE DELLE   *
  40 REM *       VENDITE       *
  50 REM *                     *
  60 REM *       SISTEMA       *
  70 REM *  EXPON. SMOOTHING   *
  80 REM ***********************
 120 GOTO 5000
 350 CLS
 352 PRINT "PREGO DIGITARE CODIC
E PRODOTTO"
 354 INPUT P$
 355 CLS
 356 PRINT " CODICE PROD.=";P$
 358 LPRINT "CODICE PROD.=";P$
 380 PRINT AT 17,0;"->PREGO DIGI
TARE DATI DI VENDITA 12 MESI PRE
CEDENTI(MAX 3 CIFRE) DA M1 A M12
"
 385 PRINT AT 1,0;" "
 390 LPRINT
 400 DIM V(1,12)
 420 FOR K=1 TO 12
 430 INPUT V(1,K)
 435 IF V(1,K)>999 THEN GOTO 200
4
 440 PRINT "M ";K;TAB (5*1);V(1,
K)
 450 LPRINT "M ";K;TAB (5*1);V(1
,K)
 460 NEXT K
 465 PAUSE 150
 470 CLS
 480 LET Y=99999999
 490 LET J=0
 495 LET A=0
 500 GOSUB 7000
 510 LET A=A+.02
 520 FOR R=1 TO 12
 530 GOSUB 8000
 540 GOSUB 8500
 550 NEXT R
 560 IF A>.980 THEN GOTO 600
 565 LET J=INT (D/12)
 570 IF J>Y THEN GOTO 600
 580 LET Y=J
 590 GOTO 500
 600 LET A=A-.02
 620 GOSUB 7000
 630 FOR R=1 TO 12
 640 GOSUB 8000
 650 GOSUB 8500
 660 NEXT R
 665 PRINT " COD.PROD.";P$
 670 GOSUB 4500
 680 PRINT "TOT=";VAP;TAB (13*1)
;"MEDIA=";INT (VAP/12*100)/100
 690 LPRINT "TOT=";VAP;TAB (13*1
);"MEDIA=";INT (VAP/12*100)/100
 740 PRINT "****->ALFA/OTT=";A
 750 LPRINT "****->ALFA/OTT=";A
 760 PRINT "DELTA/MIN=";INT (D/1
2*100)/100;TAB (18*1);"MT=";INT
MT
 770 LPRINT "DELTA/MIN=";INT (D/
12*100)/100;TAB (18*1);"MT=";INT
 MT
 830 PRINT "MT1=";INT MT1;TAB (1
2*1);"MT2=";INT MT2;TAB (22*1);"
MT3=";INT MT3
 840 LPRINT "MT1=";INT MT1;TAB (
12*1);"MT2=";INT MT2;TAB (22*1);
"MT3=";INT MT3
 880 PRINT "TREND/T=";T;TAB (16*
1);"QT=";QT
 890 LPRINT "TREND/T=";T;TAB (16
*1);"QT=";QT
 900 PRINT
 940 PRINT "*->PREV.MESE SUCC.**
**->=";INT XTP
 950 LPRINT "*->PREV.MESE SUCC.*
***->=";INT XTP
 960 STOP
 970 GOTO 5000
1000 CLS
1010 FOR I=2 TO 12
1020 LET V(1,I-1)=V(1,I)
1030 NEXT I
1040 PRINT "PREGO DIGITARE DATO
VENDITA MESE PRECEDENTE"
1050 PRINT
1055 LPRINT "COD/PROD.=";P$
1060 INPUT V(1,12)
```

Premessa

L'uomo ha sempre cercato fin dai primordi della sua lunga e travagliata esistenza, di "prevedere" e quindi "prevenire" il futuro, di capire e svelare, il divenire del tempo, il succedersi delle situazioni ambientali, il comportamento della natura, ecc.

A questo scopo e per questo scopo ha sempre dedicato parte del suo tempo e della sua intelligenza per cercare di trovare delle soluzioni che soddisfassero questa sua innata bramosia di svelare il mistero del ... domani.

Oggi, come ieri, diventa molto importante, se non determinante, ai fini aziendali e per qualsiasi impresa piccola o grossa che sia, sapere in anticipo e con sufficiente margine di attendibilità, come andranno le vendite di ogni singolo prodotto il prossimo mese o le prossime settimane.

Gli studi condotti ed intrapresi per risolvere il problema delle previsioni aziendali sono moltissimi, lo dimostra la quantità di pubblicazioni e di testi oggi esistenti in tutto il mondo su questo argomento.

Con l'utilizzo degli elaboratori elettronici si è potuta dare soluzione al problema specialmente nelle medie-grandi imprese, mentre con l'avvento del personal computer oggi è possibile dotare anche le piccole e piccolissime aziende di questo importante strumento di previsione, oltre naturalmente al fatto di poter avvicinare a questo tipo di problemi una massa enorme di persone prima escluse dal rapporto diretto con l'elaborazione elettronica dei dati.

Il programma

Il programma si basa sul principio dell'EXPONENTIAL SMOOTHING messo a punto da R.G. BROWN nel 1959 e ripreso da R. LEWANDOWSI in una recente pubblicazione alla quale rimandiamo per l'approfondimento teorico della materia (R. LEWANDOWSKI "MODELLI DI PREVISIONE PER LA PIANIFICAZIONE E LA STRATEGIA AZIENDALE" ETAS KOMPAS 1980).

Il principio dell'Exponential Smoothing o smussamento esponenziale non è altro che un caso particolare del metodo della media mobile ponderata e si basa su un semplice metodo di calcolo iterativo.

Detto principio è il seguente:

$$Xt = \alpha \cdot Xt + (1 - \alpha) \cdot Xt - 1$$

dove: Xt = rappresenta il valore della serie storica nel periodo t.

Xt = è la media calcolata

Modello di previsione delle vendite con lo ZX 80/81

della serie storica nel periodo t.
Xt−1 = è la media della serie storica del periodo precedente t−1 calcolata con lo stesso metodo.
α = è il coefficiente di ponderazione (parametro di smussamento compreso tra 0 e 1).
Il calcolo iterativo dei valori di Xt offre il vantaggio (una volta definito il valore del coefficiente di smussamento α) di definire l'intera struttura di ponderazione delle informazioni precedenti, ed è questo il principio sul quale si basa l'elaborazione della previsione avendo trovato con questo sistema la possibilità di incamerare in un unico valore numerico gli alterni andamenti dei valori precedenti.
La bontà quindi del metodo di smussamento esponenziale dipende dalla definizione del coefficiente di smussamento α.
Utilizzando i principi sommariamente più sopra descritti si cerca, con il metodo dell'analisi di una qualsiasi serie storica, di fare delle previsioni con diversi valori di α e poi confrontare dette previsioni con i valori reali della serie in esame, ottenendo in tal modo degli scarti o degli errori di previsione.
Quando la somma di questi errori di previsione risulta essere la minima di tutte quelle riscontrate durante il calcolo, allora si è in presenza del valore ottimale di alfa.
È proprio questo il metodo previsto e da noi usato per ricercare il valore ottimale del coefficiente di smussamento alfa.
Non si vuole avere la presunzione e nemmeno la pretesa che queste poche righe di commento siano sufficienti a far capire perfettamente, a chi è digiuno di questa materia, la complessità matematica del metodo, ma soltanto dare un accenno ai principi sul quale è basato.
Di sicuro vi è che il metodo è ormai diffuso e utilizzato dalle più importanti aziende nazionali ed estere per la messa a punto dei loro programmi di sviluppo con naturalmente delle sofisticazioni elaborative molto complesse definite "metodi cibernetici di autoadattamento", ma il principio in definitiva su cui si basano è pur sempre quello più sopra descritto.
Quindi per concludere, questo che presentiamo è oggi l'unico metodo finora conosciuto per fare delle ottime previsioni.
Si è volutamente utilizzato il più piccolo ed il più economico personal computer oggi esistente, lo ZX80/81 (ROM 8 K e RAM 4 K) per permettere *a tutti*, "studenti compresi", di avvicinarsi senza troppi sacrifici all'elaborazione elettronica e ad un suo utilizzo pratico e/o di ricerca che non fosse quello dei giochi.
Il programma prende in esame un prodotto per volta (per ovvie ragioni hardware e di memoria disponibile) e, analizzando i dati di vendita (introdotti da tastiera) degli ultimi dodici mesi, ricerca il valore ottimale del "coefficiente alfa" necessario per elaborare poi di seguito la previsione di vendita del periodo successivo.
Con lo ZX 80/81 si potrà (memorizzando su cassetta magnetica tanti programmi opportunamente numerati e comprensivi dei dati di vendita già introdotti quanti sono gli articoli o i prodotti da trattare e/o elaborare) rendere dinamico il lavoro di ricerca dell'"alfa ottimale" introducendo successivamente da tastiera solo il dato riguardante le vendite dell'ultimo mese e quindi rendendo possibile anche l'elaborazione dinamica dei dati storici, senza dovere, ogni volta e per ogni articolo, ricaricare i dati degli ultimi 12 mesi.
Il listato a nostro avviso può essere utilizzato come base di studio e/o di lavoro, per adattarlo a macchine diverse più complete e/o più complesse e utilizzanti memorie di massa come floppy disk ecc.
Nel nostro programma viene utilizzato lo ZX PRINTER per la stampa degli output, chi però non disponesse di tale mezzo dovrà eliminare e/o non considerare tutte le righe del listato dove compare l'istruzione "LPRINT".

```
1070 GOSUB 4500
1080 GOSUB 4600
1100 PRINT AT 14,10;"<-*NEW M12"
1110 PAUSE 100
1120 GOTO 470
2000 CLS
2002 IF V(1,12)<>0 THEN GOTO 201
0
2004 PRINT "DATI TAB.ERRATI ->RI
PARTIRE"
2006 PAUSE 150
2008 GOTO 5000
2010 PRINT "DIGITARE VALORE FISS
O DI ALFA"
2030 INPUT A
2031 CLS
2032 PRINT "ALFA/FISSO=";A;
2034 LPRINT "ALFA/FISSO=";A;TAB
(16*1);"COD/PR.=";P$
2050 GOSUB 4600
2080 GOTO 620
3000 CLS
3010 LIST 10
4000 STOP
4010 SAVE "PREV/■"
4020 GOTO 5000
4500 FOR I=1 TO 12
4510 PRINT "M ";I;TAB (5*1);V(1,
I)
4520 NEXT I
4530 RETURN
4600 FOR P=1 TO 12
4610 LPRINT "M ";P;TAB (5*1);V(1
,P)
4620 NEXT P
4630 RETURN
5000 CLS
5130 PRINT "  PREGO DIGITARE:"
5140 PRINT
5150 PRINT "->1 PER RUN INIZIO"
5160 PRINT "->2 PER CICLO DINAM"
5170 PRINT "->3 PER ALFA FISSO"
5180 PRINT "->4 PER LISTATO PR."
5190 PRINT "->5 PER REGISTRARE"
5200 INPUT W$
5210 IF W$="1" THEN GOTO 350
5220 IF W$="2" THEN GOTO 1000
5230 IF W$="3" THEN GOTO 2000
5240 IF W$="4" THEN GOTO 3000
5250 IF W$="5" THEN GOTO 4000
5300 GOTO 5000
7000 LET M1=V(1,1)
7010 LET M2=M1
7020 LET M3=M1
7030 LET XTP=M1
7040 LET VAP=0
7050 LET D=0
7060 RETURN
8000 LET X=V(1,R)
8010 LET MT1=A*X+(1-A)*M1
8020 LET MT2=A*MT1+(1-A)*M2
8030 LET MT3=A*MT2+(1-A)*M3
8040 LET MT=3*(MT1-MT2)+MT3
8050 LET T=A/(2*(1-A)**2)*((6-5*
A)*MT1-2*(5-4*A)*MT2+(4-3*A)*MT3
)
8060 LET QT=(A*A)/(2*(1-A)**2)*(
MT1-2*MT2+MT3)*(2)
8070 LET T=INT (T*100)/100
8080 LET QT=INT (QT*100)/100
8090 LET E=XTP-X
8100 LET XT1=MT+T+QT
8150 RETURN
8500 LET E1=INT E*INT E
8510 LET D=D+E1
8520 LET VAP=VAP+X
8530 LET M1=MT1
8540 LET M2=MT2
8550 LET M3=MT3
8560 LET XTP=XT1
8600 RETURN
```

Gli Input

Vengono richiesti da tastiera e contemporaneamente stampati i dodici dati riguardanti le vendite reali dei me-

Modello di previsione delle vendite con lo ZX 80/81

si precedenti. È indispensabile per una corretta ricerca dell'"alfa ottimale", digitare la storia delle vendite di almeno dodici periodi precedenti.
Nel caso di prodotti nuovi da immettere sul mercato per i quali non si dispone di tali dati basterà introdurre ripetuta per i soli ultimi 3 mesi, la previsione minima di vendita prevista per tali articoli, in modo da originare un minimo di storia necessaria per i periodi successivi.
Viene inoltre richiesto per ogni mese (di vendite) un massimo di 3 cifre, e questo può essere visto come una limitazione. Ma motivi tecnici, del mezzo a disposizione e di ottimizzazione dei tempi di elaborazione ci hanno convinto ad imporre questa regola.
D'altronde 3 cifre possono benissimo rappresentare cifre maggiori di dieci volte con uno scarto medio di 5 unità o cifre cento volte maggiori con uno scarto medio di 50 unita, e nelle previsioni questi scarti (per singolo prodotto) non sono significativi e pertanto non determinanti ai fini della validità del metodo previsionale.
Nel caso si voglia fare delle previsioni con un "coefficiente alfa fisso" o diversamente predeterminato ciò è possibile, semprechè siano presenti in memoria i dodici dati dei periodi precedenti.
Nel caso invece che si desideri rendere dinamico il sistema previsionale, bisognerà dopo avere memorizzato ogni mese su cassetta magnetica con la funzione n° 5 dati e programma, ricaricare il mese dopo il programma ed introdurre da tastiera esclusivamente il valore reale delle vendite del mese appena trascorso.
Il programma in questo caso shifterà di un mese tutti i dati tabellati perdendo il più vecchio ed incamerando il nuovo da tastiera. Con questa operazione si ottiene un continuo aggiornamento del valore ottimale del "coefficiente alfa", permettendo in tal modo di elaborare le previsioni su concetti dinamici come fatto dai grandi elaboratori gestionali.
Il tempo di elaborazione con lo ZX 80 è abbastanza lungo e può variare da un minimo di un minuto ad un massimo di 4 minuti in dipendenza dei valori mensili introdotti e dalla loro variabilità o costanza temporale, comunque volendo accelerare il processo elaborativo a scapito di una maggiore precisione basterà variare il valore di riga 510 portandolo ad esempio a 0.5.

Gli output

Gli output ottenuti sia su video che a mezzo della stampante "ZX PRINTER" prevedono l'elenco per ogni prodotto dei 12 mesi di vendita sui quali il programma fa riferimento per la ricerca del "coefficiente alfa ottimale" ed inoltre:

— Il totale delle vendite dei 12 mesi in esame;
— La media delle vendite;
— Il valore del "coefficiente alfa ottimale" ricercato;
— Il valore minimo di "DELTA" (*somma dei quadrati degli errori di previsione*);
— Il valore di MT quale media della serie storica del periodo in esame;
— Il valore di MT1 quale valore della media smussata semplice;
— Il valore di MT2 quale valore della media smussata doppia;
— Il valore di MT3 quale valore della media smussata tripla;
— Il valore del TREND lineare;
— Il valore del TREND quadratico;
— la previsione per il mese successivo.

I dati citati possono essere ottenuti:
— Come prima serie elaborata da dati appena immessi (funzione 1);
— Come serie di dati in situazione di "storia dinamica delle vendite" (funzione 2);

Modello di previsione delle vendite con lo ZX 80/81

— Con l'utilizzo di un coefficiente ALFA fisso alle volte utile per ragioni di studio e di ricerca (funzione 3).
Il programma inoltre prevede la funzione 4 che permette di avere a disposizione il listato delle istruzioni per ogni eventuale necessità di modifica e/o correzione, che per ragioni di studio si rendessero necessarie.
Mentre la funzione 5 permette, come già accennato, la registrazione su cassetta magnetica del programma e dei relativi dati immessi ed elaborati.
È questa la funzione che rende possibile il sistema dinamico della previsione e pertanto è su questa funzione che richiamiamo l'attenzione, ricordando che basterà numerare opportunamente l'istruzione 4010 per avere la possibilità di memorizzare circa una cinquantina di "programmi articoli" corrispondenti ad altrettanti prodotti dei quali si desidera prevederne in continuo l'andamento "nel tempo a venire" delle loro vendite.
Tenere sotto controllo "previsionale" circa cinquanta prodotti vuol dire pianificare e programmare la produzione e gli acquisti di quel gruppo più importante di articoli che generalmente rappresentano circa l'80% di tutto il fatturato aziendale e con una spesa insignificante "praticamente nulla" se paragonata a quella sostenuta dalle aziende più importanti per ottenere in definitiva... lo stesso scopo.

Il flow chart

In fig. 1 e 2 sono rappresentati i FLOW CHART del programma, essi sono molto semplici e non dovrebbero rappresentare una difficoltà interpretativa per chi volesse con pazienza seguirne lo sviluppo logico.
Si è voluto evidenziare graficamente le varie funzioni FOR presenti nel programma per rendere più chiaro e comprensibile lo svolgi-

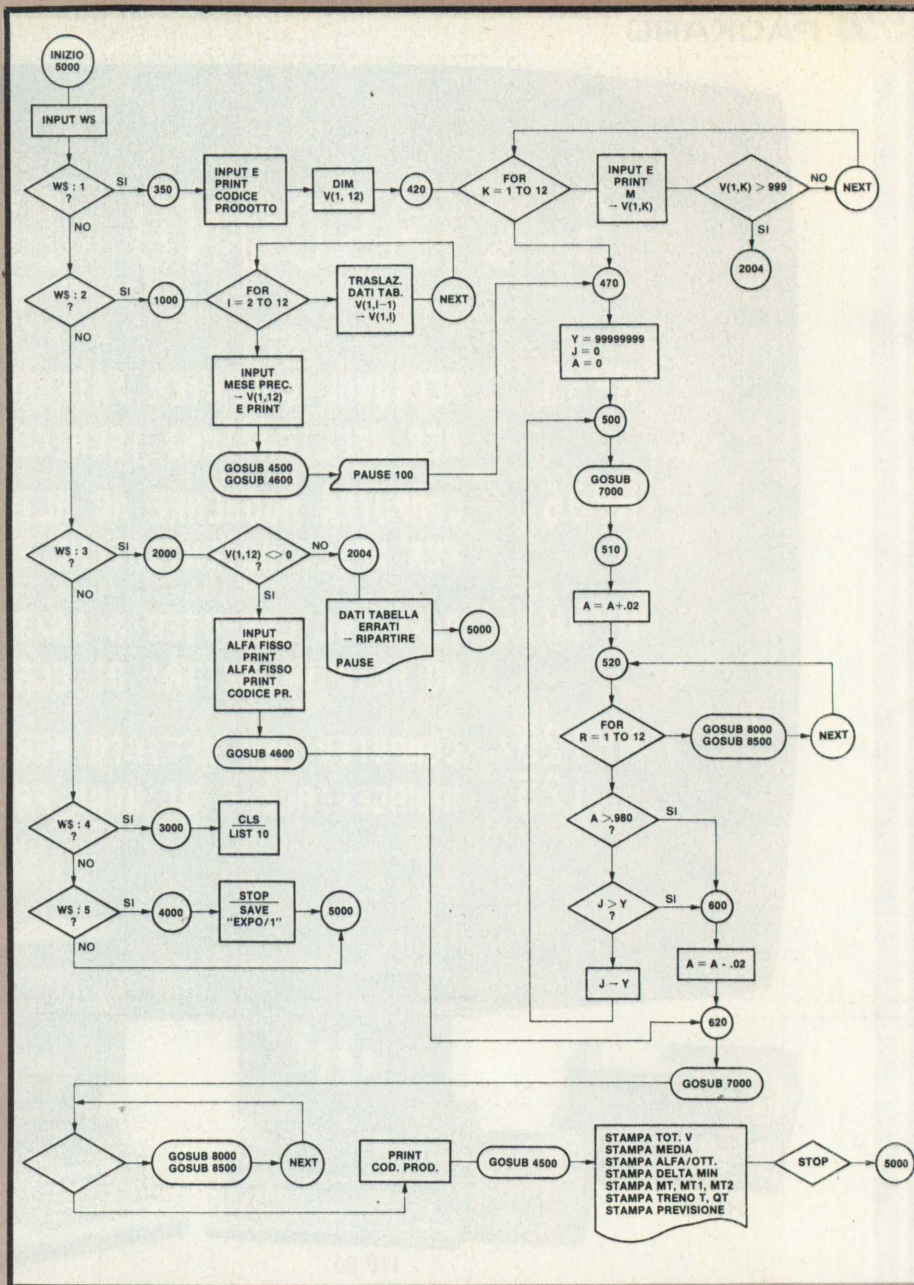



Figura 1 - Diagramma di flusso generale.

Figura 2 - Subroutine.

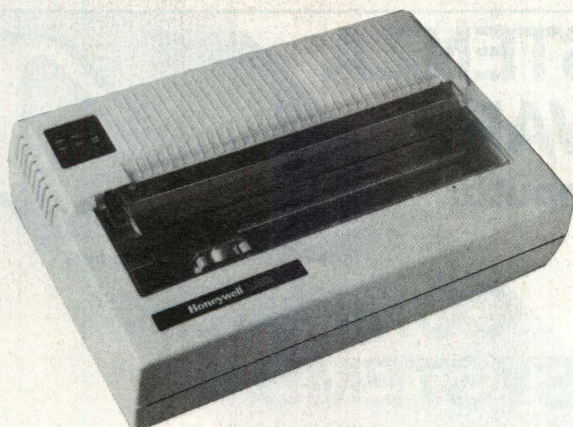
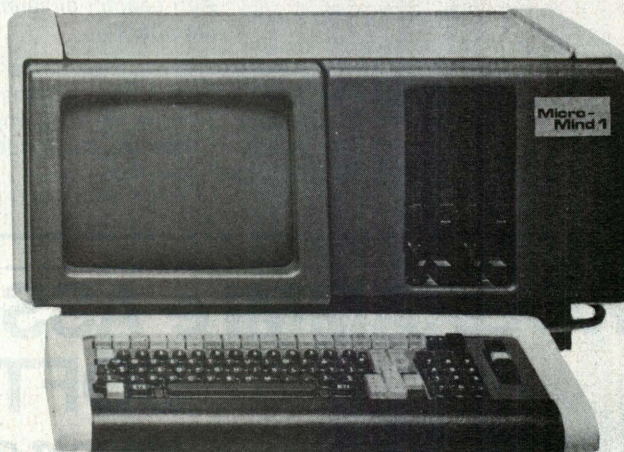
mento dell'elaborazione, ed inoltre evidenziare in appositi cerchietti il numero di quelle linee di programma interessate da dei GOTO o da nodi logici di rimando. Siccome il programma fa uso di diversi momenti dei GOSUB, abbiamo ritenuto valido il criterio di evidenziare separatamente le relative SUBROUTINE (fedeli al principio di essere il più chiari possibile).

Il listato

Anche per quanto riguarda il listato non vi dovrebbero essere difficoltà di lettura essendo molto semplici e in linguaggio versione BASIC STANDARD ZX 80/81.
Ci preme sottolineare che probabilmente un attento lettore di detto listato potrebbe a suo giudizio rivederlo ed elencarlo in modo

```
CODICE PROD.=LA/101

M 1   100
M 2   109
M 3   121
M 4   143
M 5   179
M 6   226
M 7   271
M 8   317
M 9   343
M 10  367
M 11  397
M 12  421
TOT=2994        MEDIA=249.5
****->ALFA/OTT=0.68
DELTA/MIN=127.83   MT=421
MT1=408      MT2=396      MT3=383
TREND/T=23.34     QT=-2.34
*->PREV.MESE  SUCC.****->=442


CODICE PROD.=LS/26 A

M 1   100
M 2   102
M 3   104
M 4   106
M 5   108
M 6   110
M 7   112
M 8   114
M 9   116
M 10  118
M 11  120
M 12  122
TOT=1332        MEDIA=111
****->ALFA/OTT=0.94
DELTA/MIN=0.91     MT=121
MT1=121      MT2=121      MT3=121
TREND/T=2          QT=0
*->PREV.MESE  SUCC.****->=123
```

Esempi relativi agli output del programma.

migliore e forse in maniera più ottimizzata ed è appunto questo che vorremmo ottenere perchè vorrebbe dire che la cosa che abbiamo presentato lo ha effettivamente interessato.
A commento di detto elenco di istruzioni possiamo dire che:
— La variabile V dimensionata a linea 400 conterrà i 12 valori di vendite introdotti da tastiera
— Dalla linea/riga 500 àlla linea 550, il programma esegue il calcolo della previsione per ogni valore di alfa.
All'inizio alfa viene posto al valore minimo (020) e poi successivamente incrementato di un valore identico fino a che il valore di D/12 (delta minimo) non risulti inferiore a tutti i precedenti.
La subroutine 7000 è interessata all'inizializzazione dei dati prima di

Modello di previsione delle vendite con lo ZX 80/81

eseguire il vero e proprio calcolo previsionale che sarà demandato alle subroutine 8000 e 8500.

— La linea 560 controlla che il valore di alfa non superi l'unità dovendo rimanere detto valore compreso tra zero ed uno, perchè un alfa uguale ad uno non darebbe origine ad alcuna previsione, così come un valore uguale a zero.

— La linea 570 controlla il valore minimo di delta (somma dei quadrati degli errori di previsione). Quando questo minimo viene raggiunto si deve per precisione del calcolo tornare indietro di un ciclo rifacendo il calcolo con alfa diminuito di 020 altrimenti si darebbe un alfa maggiorato in eccesso.
A questo provvedono le linee da 600 a 650.
Si è preferito adottare questa soluzione anzichè memorizzare tutti i dati del ciclo precedente, comunque, non è detto che con la menzionata memorizzazione si usi più memoria di quanta da noi sprecata con il nostro metodo. Si può a questo proposito provare con un esperimento in tal senso.

```
ALFA/FISSO=0.2   COD/PR.=LA/101
M 1   100
M 2   109
M 3   121
M 4   143
M 5   179
M 6   226
M 7   271
M 8   317
M 9   343
M 10  367
M 11  397
M 12  421
TOT=2994         MEDIA=249.5
****->ALFA/OTT=0.2
DELTA/MIN=1068.5  MT=432
MT1=311        MT2=222    MT3=163
TREND/T=38.64    QT=1.9
*->PREV.MESE SUCC.****->=472


COD/PROD.=LA/101
M 1   121
M 2   143
M 3   179
M 4   226
M 5   271
M 6   317
M 7   343
M 8   367
M 9   397
M 10  421
M 11  455
M 12  468
TOT=3708         MEDIA=309
****->ALFA/OTT=0.68
DELTA/MIN=172       MT=468
MT1=459       MT2=448     MT3=437
TREND/T=15.68    QT=-4.64
*->PREV.MESE SUCC.****->=479
```

Altri esempi di valori che vengono elencati su video o su stampante.

— Dalla linea 665 alla linea 960 il programma si occupa delle stampe dei dati sia per il video che per il printer.

— Dalla linea 1000 a linea 1120, il programma esegue lo shiftamento degli 11 dati precedenti per far posto all'ultimo dato di vendita del mese appena trascorso, e che verrà richiesto da tastiera (linee 1040 - 1060) per poi rimandare il programma a linea 470 per il completo ricalcolo dell'alfa avendo *la tabella dati*, perso il dato più vecchio ed incamerato quello più recente.
È questa l'operazione del cosiddetto CICLO DINAMICO di funzione 2.

— Da linea 2000 a linea 2080 viene eseguita l'accettazione di un alfa fisso richiesto da tastiera previo controllo della presenza di dati nel "vettore dati V" per poi rimandare alla linea 620 per l'esecuzione del calcolo previsionale con alfa fisso.

Pensiamo di avere descritto e documentato il programma nel modo più completo possibile con la speranza di essere stati anche chiari e sufficientemente concisi.
Per ultima cosa presentiamo una serie di tabulati dove si potrà verificare la validità del nostro sistema previsionale, sulla base di diverse situazioni storiche (con codici prodotti diversi), nonchè altri ottenuti in situazione dinamica (con codice prodotto uguale) ed uno con alfa predeterminato.

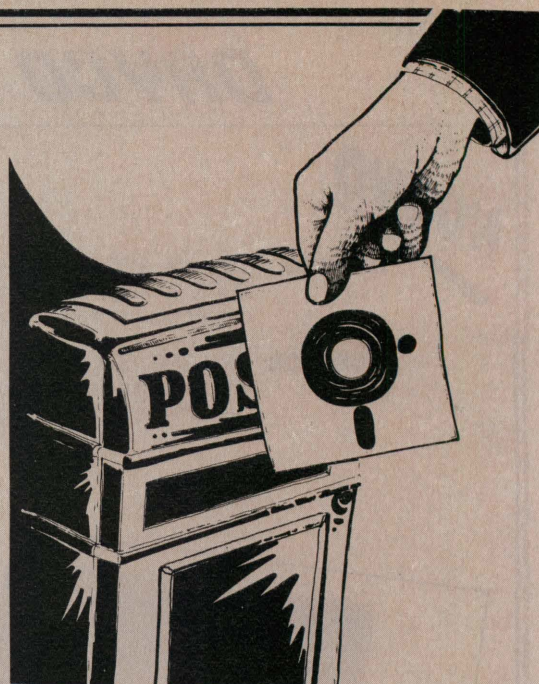
Per collaborare ... meglio!
tutti i lettori che desiderano collaborare a "Riservato Personal" devono inviare una dettagliata descrizione del programma, il relativo listato (possibilmente stampato su carta bianca) e il supporto magnetico (disco o cassetta) a:

"Riservato Personal"

Via Rosellini, 12
20124 MILANO

Sono anche disponibili le specifiche dettagliate per la preparazione degli articoli. Richiedetele allo stesso indirizzo.

SERVIZIO SOFTWARE

Bit propone ai propri lettori i dischi o le cassette dei migliori programmi pubblicati. I programmi, provati e garantiti, sono di immediato utilizzo.

Bit n°	Programma	Sistema	Prezzo	Codice	Supporto
38	Gioco della scimmia Spaccamattoni	VIC 20	15.000	VI381A	Cassetta
38	Text-Editor	PET3032	20.000	PE381B PE382B	Cassetta Disco
38	Planel	Apple II	20.000	AP382C	Disco
39	Rompicapo di Rubik	CBM 4032 CBM 3032	15.000 20.000	PE391A PE392B	Cassetta Disco
39	Breakout	CBM 3032	20.000	PE393A	Cassetta
40	Reporter	Apple II	20.000	AP402C	Disco

Per richiedere i programmi in contrassegno, pagando direttamente al postino la cifra indicata + L. 2.000 quale contributo fisso per spese di spedizione, inviare il seguente tagliando

Spedire in busta chiusa a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Inviatemi i seguenti nastri e/o dischi con i programmi pubblicati su Bit.

GRUPPO EDITORIALE JACKSON

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Spese postali (contributo fisso) L. 2.000

TOTALE L. ....

che pagherò al postino alla consegna del pacco.

Cognome ....

Nome ....

Indirizzo ....

CAP .... **Città** ....

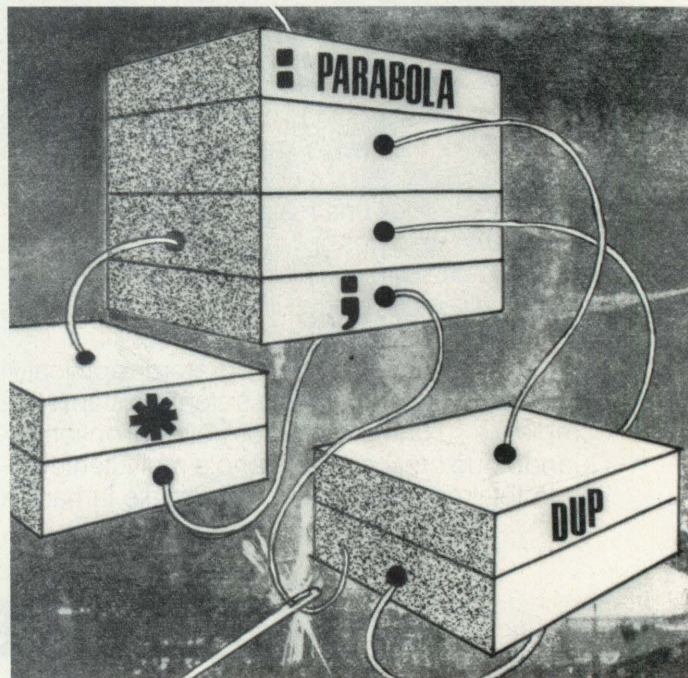
Firma ....

di **M. Tausel**

Seconda Parte

Nella precedente introduzione al mondo per molti fino a poco tempo fa inesplorato e un poco misterioso del FORTH se ne sono messi in risalto i fondamenti: l'architettura dello stack, l'organizzazione a dizionario tipica del linguaggio, le strutture di controllo del flusso delle istruzioni. In questa seconda parte saranno esaminate altre possibilità che il FORTH offre, quali l'inserimento di segmenti redatti in linguaggio macchina (assembler), la gestione di uno schema di memoria virtuale e l'utilizzo delle funzioni editoriali più comuni.

Le routine in assembler

Il processo di ricerca delle definizioni usate nella creazione di una certa word richiede un tempo più o meno lungo. Talvolta la velocità costituisce un fattore critico, mentre in altri casi si presenta la necessità di scrivere un programma "driver" per interfacciare un nuovo dispositivo di I/O. Per tali evenienze il sistema FORTH mette a disposizione un Assembler per la definizione di routine in codice macchina.
Tali routine possono essere usate esattamente come ogni altra definizione FORTH. L'Assembler FORTH adotta anch'esso la sintassi postfissa già vista, in base alla quale si inseriscono prima operandi e modi di indirizzamento, seguiti dai codici operativi (mnemonici). Gli operandi e i modi di indirizzamento inseriscono dati o indirizzi sullo stack. Il codice operativo combina questi argomenti con il codice d'istruzione binario e inserisce l'istruzione completa nel dizionario dove è in corso di compilazione la definizione della nuova operazione. L'Assembler traduce il codice sorgente in una passata e non usa alcuna tabella dei simboli. Lo stack viene utilizzato per memorizzare gli indirizzi degli operandi forniti da calcoli precedenti oppure dalle variabili e dalle costanti del FORTH (figura 1).
L'Assembler è dotato di un proprio corredo di istruzioni condizionali, comprese IF... THEN e BEGIN... END.
Queste operazioni creano istruzioni di test e salto condizionato nel linguaggio della macchina utilizzata. Le corrispondenti definizioni ad alto livello (vocabolario FORTH) sono tenute separate poichè risiedono in un differente vocabolario. Qualcuno qui si chiederà come possa un Assembler per giunta ad una sola passata, gestire salti di lunghezza sconosciuta come ad esempio nei costrutti di tipo IF... THEN. Il "miracolo" si spiega col fatto che l'IF assembla una istruzione di salto condizionato nella routine, lasciando in bianco l'indirizzo sconosciuto, inoltre inserisce la locazione dell'indirizzo in bianco sullo stack (lo stesso utilizzato normalmente dal FORTH).
Quando viene raggiunto THEN, l'indirizzo di arrivo del salto è conosciuto, quindi l'indirizzo lasciato in bianco viene rimpiazzato dal valore corretto.

La memoria virtuale

L'organizzazione del FORTH prevede anche uno schema di memoria virtuale. Suddivisa in blocchi, può contenere i codici sorgenti dei programmi applicativi e tabelle di dati.
Ricordiamo che il termine "me-

```
HEX OK
CODE START   A SUB   1E OUT   NEXT JMP OK
CODE STOP   1 A MVI   1E OUT  NEXT JMP OK
DECIMAL OK

: DELAY   0 DO   LOOP ; OK

5000 DELAY OK

: CYCLE   START DUP DELAY  STOP DELAY ; OK
: TEST   50 0 DO  I  PARABOLA2 CYCLE LOOP ; OK

START OK
STOP OK

10 CYCLE OK
TEST OK
```

Figura 1 - Il linguaggio Assembler del FORTH è mischiato con definizioni in FORTH stesso, ad esempio per il controllo dell'input/output.
La definizione FORTH di START e STOP usano il codice macchina dell'8080 per commutare un port di uscita. DELAY è una routine di ritardo scritta in FORTH che preleva dallo stack un argomento proporzionale al ritardo. CYCLE accetta un argomento, un ritardo, e commuta il port di uscita per quel periodo di tempo. TEST usa una word FORTH definita in precedenza, PARABOLA 2, per calcolare una serie di cicli di commutazione del port di uscita.

Il linguaggio FORTH: potenza e semplicità

moria virtuale" descrive il processo che rende invisibile all'utente la distinzione tra memoria principale è memoria secondaria. Concettualmente è la capacità del sistema operativo di accedere e di trasferire in memoria centrale qualunque parte della memoria secondaria in maniera del tutto trasparente.
I riferimenti necessari per la corretta sistemazione in memoria centrale (*delle parti giuste al momento giusto da/verso memoria di massa* ndr) sono un compito del sistema operativo. Quello del FORTH implementa la gestione di memoria virtuale assumendo come segmenti di "swap" zone numerate di disco (o di altre memoria di massa) lunghi solitamente 1024 byte ciascuno e chiamate blocchi (figura 2).
Questi blocchi vengono trasferiti, ogni volta che si rende necessario, dal disco in un buffer (o più di uno) composto da 1024 byte di dati preceduti da 2 byte contenenti informazioni relative al blocco e seguiti da 2 byte che fungono da limitatori. Per accedere ad un blocco viene usato il comando:

n BLOCK

dove "n" è il numero logico del blocco. BLOCK inserisce sempre l'indirizzo del primo byte del blocco nello stack.
Se il blocco che viene richiesto si trova già in memoria, il trasferimento non viene materialmente eseguito per cui i blocchi che vengono usati più di frequente vengono mantenuti in memoria a lungo, onde i trasferimenti da disco vengono minimizzati in modo statisticamente automatico.

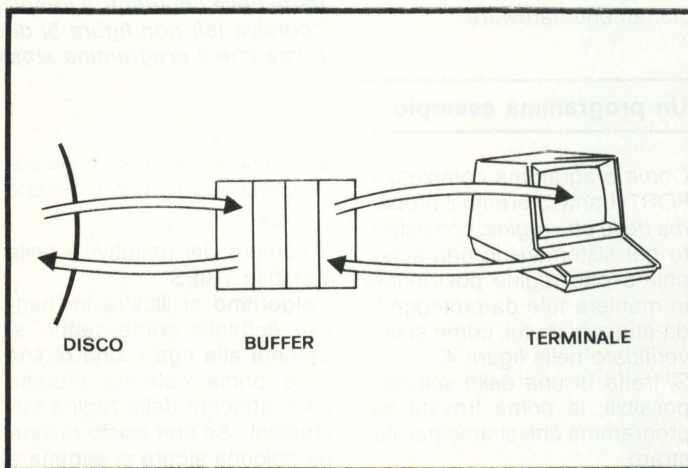



Figura 2 - Parti della memoria secondaria (blocchi) vengono trasferite nei buffer quando necessario in maniera trasparente all'utente. Il sistema operativo del FORTH gestisce completamente questo sistema chiamato "memoria virtuale".

```
SCR # 16                                                    TLS23-MAR-82 )
  0 ( ESEMPI DI RICURSIONE
  1 : MYSELF    ( PERMETTE LA CHIAMATA RICORSIVA DI UNA WORD )
  2           LATEST PFA CFA , ; IMMEDIATE
  3 : FATT    ( n --- fatt  FATTORIALE DI UN NUMERO <= 7 )
  4           -DUP IF DUP 1 - MYSELF * ELSE 1 ENDIF ;
  5 ;S
  6
  7
  8
  9
 10
 11
 12
 13
 14
 15
```

Figura 3 - Esempio di come viene rappresentato in FORTH un blocco di 1024 caratteri di codice sorgente. In questo esempio si vede quanto sia facile implementare funzioni ricorsive utilizzando la word MYSELF.

I blocchi sono registrati sul disco consecutivamente e numerati sequenzialmente a partire da zero, il che significa che i primi 1024 caratteri di dati presenti sul disco sono catalogati come blocco zero, il secondo gruppo come blocco 1 e via di seguito.
Quando un blocco contiene codice sorgente viene denominato convenzionalmente "schermo" (screen) ed è suddiviso in 16 linee di 64 caratteri ciascuna, in guisa tale che risulti possibile visualizzare una schermata anche su terminali di vecchio tipo (figura 3).
Essendo solitamente le definizioni brevi (raramente occupano più di 3 linee) tutto il codice di una routine viene visualizzato nella sua interezza.
Un comando LOAD preleva il numero dello schermo dallo stack e lo legge come se i 1024 caratteri fossero stati digitati direttamente dal terminale.
Lo schermo può contenere definizioni ad alto livello oppure in assembler, come operazioni da eseguirsi immediatamente.
Nel vocabolario standard FORTH sono disponibili altre word per il trattamento dei blocchi. Ad ogni buon conto, dato che come s'è visto il FORTH è un linguaggio espandibile, si ha sempre la possibilità di modificare il sistema di gestione delle memorie di massa, ad esempio implementando un sistema di gestione dei file simile a quello usato in molti altri linguaggi (BASIC, FORTRAN, ecc.).

Il linguaggio FORTH: potenza e semplicità

L'Editor

Per poter scrivere o modificare programmi su schermi, e poterli poi memorizzare su memoria di massa, è necessario richiamare un vocabolario denominato Editor. Per utilizzarlo basta digitare Editor.
Per selezionare uno schermo per l'edit, si inserisce; n EDIT, ove "n" rappresenta il numero dello schermo desiderato. I comando disponibili sono suddivisi in 4 gruppi:

- comandi per editare uno schermo;
- comandi per muovere il cursore;
- comandi per il trattamento delle stringhe;
- comandi per editare linee;

Essendo FORTH un linguaggio eminentemente interattivo, il programmatore trascorrerà un tempo assai maggiore al terminale che non a tavolino, al contrario di quanto avviene con altri linguaggi specie se compilati. In generale si scrivono alcune note relative al problema da risolvere insieme alle righe essenziali del programma. Per problemi di una certa dimensione sarà tuttavia necessario e consigliabile sviluppare la struttura del programma proposto in maggiore dettaglio secondo le usuali tecniche Top-down (programmazione discendente). Dopo di che sarà comunque subito possibile sedere al terminale, iniziando a testare per gradi la soluzione introducendo una o due definizioni, fino all'ottenimento di risultati corretti, per combinarle tra loro onde ottenere definizioni più potenti.
Per memorizzarle in modo permanente si inseriscono poi in uno schermo.
Riepilogando, anche se i programmi FORTH vengono verificati e caricati partendo dal livello più basso sino a quello più alto (bottom-up, ossia programmazione ascendente) la fase di progettazione corretta ha luogo partendo dalla definizione delle funzioni svolte dalle word del livello più alto via via scendendo (programmazione discendente o top-down). I tentativi di definire un numero di word di basso livello che si pensa possano tornare utili al fine di una loro successiva integrazione non è di per sè criticabile, però il più delle volte costituisce una maniera sicura per perdere tempo e aumentare gli sforzi (*ciò per una serie di motivi tra cui la difficoltà di definire "mattoni" realmente "tuttofare"* Ndr).
È da notare l'estrema semplicità di ogni livello del processo di sviluppo di un programma applicativo. Ogni singola operazione dovrebbe essere definita separatamente e tutto dovrebbe essere tenuto il più semplice possibile. L'esistenza di definizioni a basso livello renderà più comodo eventuali modifiche e la diagnosi di eventuali malfunzionamenti hardware.

Un programma esempio

Come programma completo in FORTH analizzeremo il *problema delle otto regine*, consistente nel sistemare in una scacchiera otto regine posizionate in maniera tale da proteggerle da attacchi mutui, come si può verificare nella figura 4.
Si tratta di una delle soluzioni possibili, la prima trovata dal programma che stiamo per illustrare.
L'algoritmo proposto è di tipo ricorsivo ed introduce una regina alla volta in righe discendenti valutando in quale casella della riga si trova al sicuro dagli attacchi di quelle presenti fino a quel momento sulla scacchiera, qualora tale posizione non esistesse ritorna alla riga precedente spostando la regina ivi posizionata e così via. Il relativo programma FORTH è in figura 5.
Vengono utilizzati quattro vettori:

- A indica con uno zero nelle caselle opportune le colonne non sicure da attacchi sulla verticale.
- B indica con uno zero nelle caselle opportune le posizioni non sicure a causa di un attacco di regina da diagonale situata nel quadrante superiore destro. Per localizzare la casella si usa la formula B (riga + colonna).
- C come in B ma per attacchi dal quadrante superiore sinistro e formula C (colonna - riga + 7).
- X contiene il numero della colonna sicura per ciascuna riga.

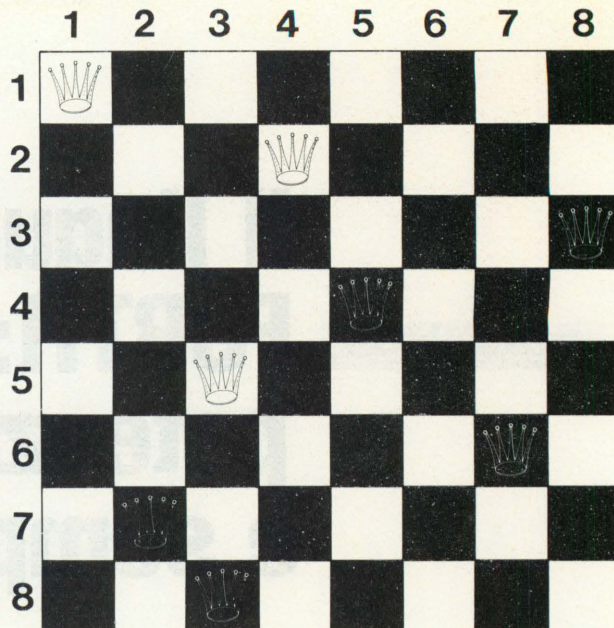



Figura 4 - Il problema delle 8 regine: debbono disporre sulla scacchiera in modo che ciascuna sia al riparo da minacce da parte delle rimanenti. Il rompicapo trova in FORTH una soluzione ricorsiva (cfr con figura 5) di cui la posizione qui illustrata è la prima che il programma stesso escogita.

Il numero dei tentativi è nella variabile TRIES.
L'**algoritmo** si illustra in maggior dettaglio come segue: si sistema alla riga n una regina nella prima colonna protetta dagli attacchi delle regine sovrastanti. Se non esiste nessuna colonna sicura si aspetta a sistemarla e si passa alla riga n-1 spostando verso destra la regina corrispondente fino alla successiva colonna protetta; se non ne esistono si passa alla riga n-2 eseguendo gli stessi passi e così via. Trovata alfine una colonna sicura si risistema la regina, si scende di una riga e si cerca di sistemare la regina ivi presente.
Una soluzione viene stampata ogni qualvolta si sistemano correttamente tutte le regine.
L'iterazione continua fino ad esaurimento di tutte le possibilità. L'algoritmo proposto non considera l'eliminazione delle soluzioni simili per riflessione e rotazione per cui le soluzioni stampate sono in numero ridondante rispetto allo stretto indispensabile.
Passiamo ora ad analizzare le singole word:
START: azzera la variabile TRIES e attiva TRY.
TRY: è la word chiave del programma: sviluppa l'algoritmo appena esposto utilizzando le word che verranno ora descritte.
PRINTSOL: stampa su una riga la scritta "PROBLEMA RISOLTO AL TENTATIVO N.", il numero del tentativo ed i numeri delle otto colonne relative alle righe nelle quali le regine sono protette.
UNMARK: riga col ——— (preleva dallo stack i due valori) riporta a 1 le caselle dei vettori A, B, C cioè A(col), B(riga+col), C(col-riga+7)
MARK: riga col ——— (preleva dallo stack i due valori) porta a zero le caselle corrispondenti dei vettori A, B, C cioè A(col), B(riga + col), C(col-riga +7)
SAFE: riga col ——— f verifica se la posizione (riga, colonna) è sicura; se lo è f è settato a uno.
IARRAY: è una word che, utilizzando la struttura <BUILDS DOES> crea vettori di lunghezza variabile riempiti di uni. In esecuzione dato il numero della cella ne restituisce l'indirizzo.
MYSELF: word che permette la creazione di strutture ricorsive.
Come si nota il programma è stato inserito negli "schermi". 18 e 19; per eseguirlo è necessario caricare gli schermi con 18 LOAD. Poi è sufficiente digitare START per mandarlo in esecuzione.
Purtroppo molte delle word usate nell'esempio risulteranno completamente sconosciute al lettore che si avvicina per la prima volta al FORTH, d'altra parte una spiegazione estensiva

```
SCR # 18                                                   TLS 15-MAR-82 )
  0 ( PROBLEMA DELLE OTTO REGINE
  1 : MYSELF   LATEST PFA CFA , ; IMMEDIATE
  2 : IARRAY   <BUILDS 0 DO 1 , LOOP  DOES>   SWAP DUP + + ;
  3  8   IARRAY A
  4  16  IARRAY B
  5  16  IARRAY C
  6  8   IARRAY X
  7 : SAFE     SWAP OVER OVER OVER OVER - 7 + C @ >R + B @ >R
  8            DROP A @ R>  R> * * ;
  9 : MARK     SWAP OVER OVER OVER OVER - 7 + C 0 SWAP !
 10            + B 0 SWAP ! DROP A 0 SWAP ! ;
 11 : UNMARK   SWAP OVER OVER OVER OVER - 7 + C 1 SWAP !
 12            + B 1 SWAP ! DROP A 1 SWAP ! ;
 13 -->
 14
 15


SCR # 19                                                   TSL 15-MAR-82 )
  0 ( PROBLEMA DELLE OTTO REGINE
  1 0 VARIABILE TRIES
  2 : PRINTSOL     ." PROBLEMA RISOLTO AL TENTATIVO N. " TRIES @ 6
  3                8 0 DO  I * @ 1 + 5 .R  LOOP CR ;
  4 : TRY   8 0 DO  1 TRIES +! ?TERMINAL IF QUIT THEN DUP I SAFE
  5         IF DUP I MARK DUP I SWAP X ! DUP
  6         7 < IF DUP 1+ ?STACK MYSELF ELSE PRINTSOL THEN
  7         DUP I UNMARK THEN LOOP DROP ;
  8 : START    0 TRIES ! 0 CR TRY ;              ;S
  9
 10
 11
 12
 13
 14
 15
```

Figura 5 - Questi due "schermi" 18 e 19 contengono il codice sorgente completo in FORTH del programma per risolvere il problema delle otto regine.

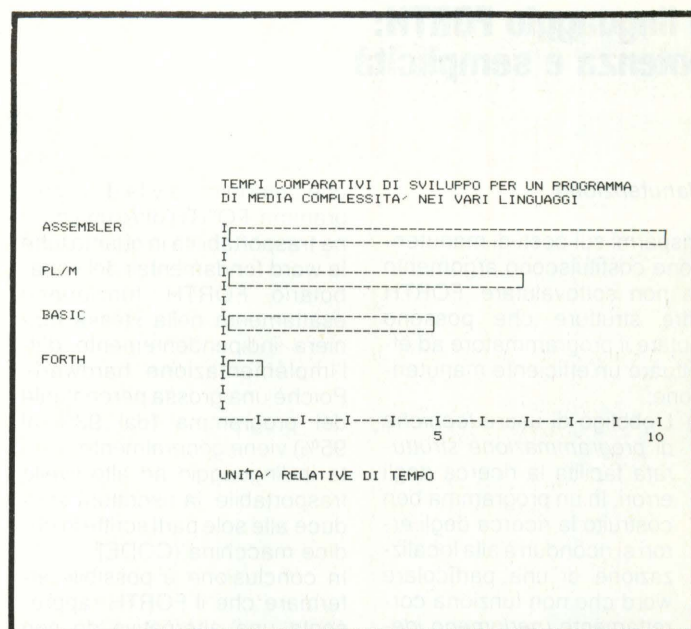



Figura 6 - Tempi comparativi di sviluppo per un programma di media complessità nei vari linguaggi.

Figura 7 - Tabella di comparazione per l'occupazione di memoria di un programma applicativo piuttosto complesso nei vari linguaggi.

del programma avrebbe reso l'articolo una sorta di manuale del FORTH.

Comparazione delle prestazioni

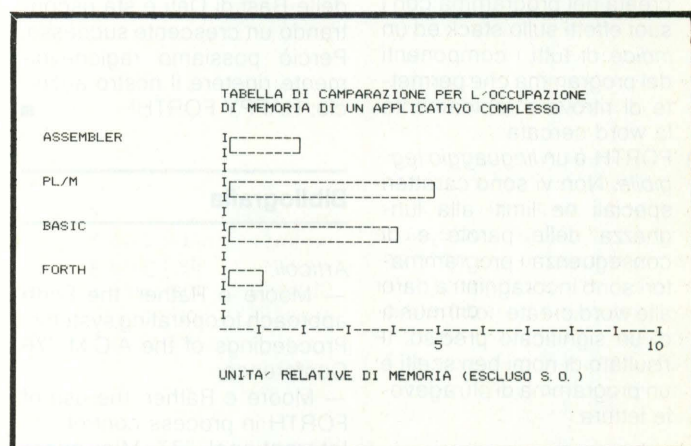
Si cercherà ora di fornire qualche elemento di confronto delle prestazioni del FORTH rispetto ad altri linguaggi in cinque aree che concorrono tipicamente a determinare i costi del software: tempo di sviluppo di un programma, occupazione di memoria, velocità di esecuzione, manutenzione, trasportabilità.

Tempo di sviluppo di un programma (diagramma di figura 6)

Gli utenti del FORTH concordano nello stimare che i tempi di sviluppo vengono ridotti di una percentuale variabile tra il 60% ed il 90%. Questi impressionanti risparmi sono il frutto di diversi pregi del FORTH:

1) La struttura a vocabolario permette di creare linguaggi dedicati con i quali il programmatore può descrivere molto più rapidamente le proprie esigenze. Si provi a spiegare a qualcuno le funzioni di un "disk controller" senza utilizzare la terminologia specializzata dell'industria!!
2) La naturale interattività del FORTH permette di provare il programma e l'hardware man mano che il lavoro procede.
3) Questo linguaggio impone l'uso di tecniche di programmazione strutturata, ciò assicura modularità nel progetto software e quindi facilita il collaudo, la documentazione e la manutenzione.
4) Essendo il FORTH un vero linguaggio a più livelli permette l'inserzione di parti in Assembler in qualunque fase dello sviluppo di un programma senza necessità di ricorrere a subroutine.
5) Un programma costruito con vocabolari definiti dall'utente è in genere più compatto. È generalmente accettato il fatto che il tempo di sviluppo di un programma è indipendentemente dal linguaggio usato, proporzionale alla dimensione del programma stesso. Un programma scritto in FORTH permette una riduzione del numero di linee da tre a dieci volte rispetto ad altri linguaggi di uso corrente.

Occupazione di memoria (cfr figura 7)

Qualora sia importante risparmiare spazio di memoria FORTH permette economie consistenti, di particolare interesse con programmi di una certa mole, ovviamente reso più vivo nel caso di sistemi dotati di capacità mnemonica non troppo estesa.

Velocità di esecuzione

Come per ogni linguaggio ad alto livello la velocità esecutiva dipende dal tempo speso per eseguire le operazioni richieste dal S.O. e dal linguaggio. Contrariamente a quel che potrebbe apparire ad un primo giudizio superficiale di chi apprende, per la prima volta, che si tratta di linguaggio interpretativo, il FORTH ne permette un controllo abbastanza efficace consentendo l'utilizzo sia di definizioni ad alto livello (piuttosto veloci nell'esecuzione essendo precompilate sotto forma di stringhe di indirizzi di word precedentemente definite) sia di definizioni di tipo "code" (eseguite alla massima velocità essendo compilate in codice macchina). In altre parole FORTH è in media piuttosto veloce: nella maggior parte delle applicazioni l'overhead è notevolmente inferiore a quello tipico di altri linguaggi; ad esempio è circa dieci volte più veloce del BASIC.

Il linguaggio FORTH: potenza e semplicità

Manutenzione

I risparmi sui costi di manutenzione costituiscono argomento da non sottovalutare. FORTH offre strutture che possono aiutare il programmatore ad effettuare un'efficiente manutenzione:

1) L'obbligo di usare *tecniche di programmazione strutturata* facilita la ricerca degli errori. In un programma ben costruito la ricerca degli errori si ricondurrà alla localizzazione di una particolare word che non funziona correttamente (*perlomeno idealmente....* Ndr)
2) La documentazione in FORTH, necessaria come negli altri linguaggi, vengono utilizzati tipicamente un *glossario* nel quale viene definita l'azione di ogni word creata nel programma con i suoi effetti sullo stack ed un *indice* di tutti i componenti del programma che permette di ritrovare rapidamente la word cercata.
3) FORTH è un *linguaggio leggibile*. Non vi sono caratteri speciali ne limiti alla lunghezza delle parole e di conseguenza i programmatori sono incoraggiati a dare alle word create nomi muniti di un significato preciso. Il risultato di nomi ben scelti è un programma di più agevole lettura.

Trasportabilità

La trasportabilità di un programma è uno dei punti-chiave che determina la vita di un programma. FORTH offre una buona trasportabilità in quanto tutte le word fondamentali del vocabolario FORTH funzionano esattamente nella stessa maniera indipendentemente dall'implementazione hardware. Poichè una grossa percentuale del programma (dal 93% al 95%) viene generalmente scritta in linguaggio ad alto livello trasportabile, la riscrittura si riduce alle sole parti scritte in codice macchina (CODE).

In conclusione è possibile affermare che il FORTH rappresenta una alternativa da non sottovalutare ai linguaggi ed ai sistemi operativi esistenti; attualmente viene applicato a una grande varietà di aree che vanno dalla acquisizione di dati on-line alla loro analisi, al controllo dei processi, alla gestione delle Basi di Dati e sta riscontrando un crescente successo. Perciò possiamo ragionevolmente ripetere il nostro auspicio: HAPPY FORTH! ■

FORTH è disponibile negli U.S.A. per quasi tutti i micro e mini computer: Alpha Micro, Apple, Atari, Cromenco, DEC PDP/LSI-11 Heath-89, Hewlett-Packard 85, 75C, IBM PC, Micropolis, North Star, Ohio Scientific, Osborne, Pet SWTPC, TRS-80 I, II, III, Color e per quelli operanti con CP/M. È persino disponibile per grandi sistemi tipo IBM 370, 4341, 3033 ecc.
Oltre alle molte società che si occupano sia di sistemi e applicazioni in FORTH che di consulenza e corsi di aggiornamento il gruppo che maggiormente si è occupato della diffusione del FORTH sia a livello professionale che hobbystico è il FORTH Interest Group (FIG) PO BOX 1105 San Carlos, CA 94070.
In Italia vari rivenditori offrono a catalogo versioni del FORTH per molti mini e microcomputer.
Quest'anno si è formato il Forth Interest Group Italia legato al FIG, che ha come finalità lo scambio di informazioni ed esperienze tra programmatori FORTH e la divulgazione del linguaggio. Sono previsti incontri periodici. Le persone interessate possono contattare Marco Tausel - Via Forni, 48 - 20161 Milano.

Bibliografia

Articoli:

— Moore e Rather, the Forth approach to operating systems. Proceedings of the A.C.M. '76 Conference.
— Moore e Rather, the use of FORTH in process control. International '77 Mini-micro computer conference, Ginevra.
— James, FORTH for microcomputers. ACM SIGPLAN ottobre 1978
— Hicks, FORTH's forte is tighter programming. Electronics marzo 1979.
— vari articoli su BYTE dell'Agosto 1980
— notiziario bimestrale del Forth Interest Group (Forth Dimensions
— vari articoli pubblicati dalla FORTH Inc.

Libri:

— Harry Katzan Jr., invitation to FORTH. Petrocell
— FORTH user's manual. Rockwell
— Rather e Brodie, Using FORTH, Forth Inc. (aprile 1981)
— Installation manual, glossary, model and editor. Forth Interest Group (novembre 1980)
— Stevens, Kitt peak Primer. reperibile da F.I.G. (1979)
—FORTH 79 Standard. Una pubblicazione del Forth Standard Team reperibile da F.I.G.
— Leo Brodie, Starting FORTH. Prentice Hall (1981)
— Thom Hogan, Discover FORTH. Osborne Mc Graw Hill (1982)
— R.G. Loeliger, Threaded Interpretative Languages. Byte Books (1981)

Nel prossimo numero:

Aquisire i dati con il VIC 20
Anatomia dello ZX 81
Il curvilineo computerizzato
L'assembly rivisitato

RISERVATO PERSONAL

di **M. Gremes**

Questo...brano costituisce semplicemente una serie di richiami di teoria musicale che certamente molti lettori conoscono. In omaggio al principio, che qui inauguriamo, di spezzare la teoria e la pratica nelle pagine di Bit "normali" ed in quelle del nuovo inserto le riproduciamo qui precisando che sono la premessa dell'articolo pratico "Polimus" dello stesso autore. Chi, ripetiamo, di musica e relativa notazione se ne intende già può saltare queste pagine, ma per tutti coloro che vogliono corredare di motivetti da copiare da un qualche spartito i loro programmi, si tratta di un piccolo vademecum che, evitando i sarchiaponi (cioè quelle bestioline che tutti danno per conosciute, senza saperne nulla) può servire in più occasioni.

La maggior parte dei personal presenti sul mercato, offrono capacità sonore più o meno sofisticate. Per quelli che ne sono sprovvisti di solito vengono fornite schede aggiuntive.

Diversi sono i sistemi usati per generare suoni: si va dal più semplice che consiste nel simulare con dei ritardi software la frequenza voluta, ricavabile poi su una linea d'uscita, al più potente che sfrutta le caratteristiche di uno o più integrati appositamente studiati.

Quali che siano i computer, le schede, i sistemi usati, ciò che più conta, ovviamente, è il massimo sfruttamento delle capacità della macchina.

Chi scrive è dell'opinione che in ogni tipo di programma sia vantaggioso sfruttare le possibilità sonore così come, nel caso siano presenti, si usano sempre più la grafica e il colore. E non è solo un orpello.

Ad esempio, in un comune programma gestionale, piacevoli motivetti opportunamente diversificati potrebbero richiamare l'attenzione dell'operatore, facendogli distinguere con chiarezza e immediatezza le varie fasi dell'elaborazione: sicuramente molto più vivo del glaciale bip, il che oltretutto non guasta se si vuol rendere il lavoro meno monotono (*se non un tantino divertente: il principio non è frivolo, ma in accordo coi moderni canoni dell'argonomia e dei "fattori umani"* NdR).

Il problema che si pone è però quello di convertire la melodia in dati espressi e codificati nella forma voluta dall'elaboratore, e visto che la musica ha un sistema di notazione proprio, è necessario essere in grado di decifrare questo particolare insieme di codici. A questo punto non ha importanza se nella macchina X il sol3 (nota Sol terza ottava) equivale ad un numero o in quella Y ad un altro, perché la corrispondenza tra nota e numero si fisserà (e modificherà) in apposite tavole. Quel che conta è invece sapere che quel cerchietto in una data posizione del pentagramma corrisponde ad una certa nota.

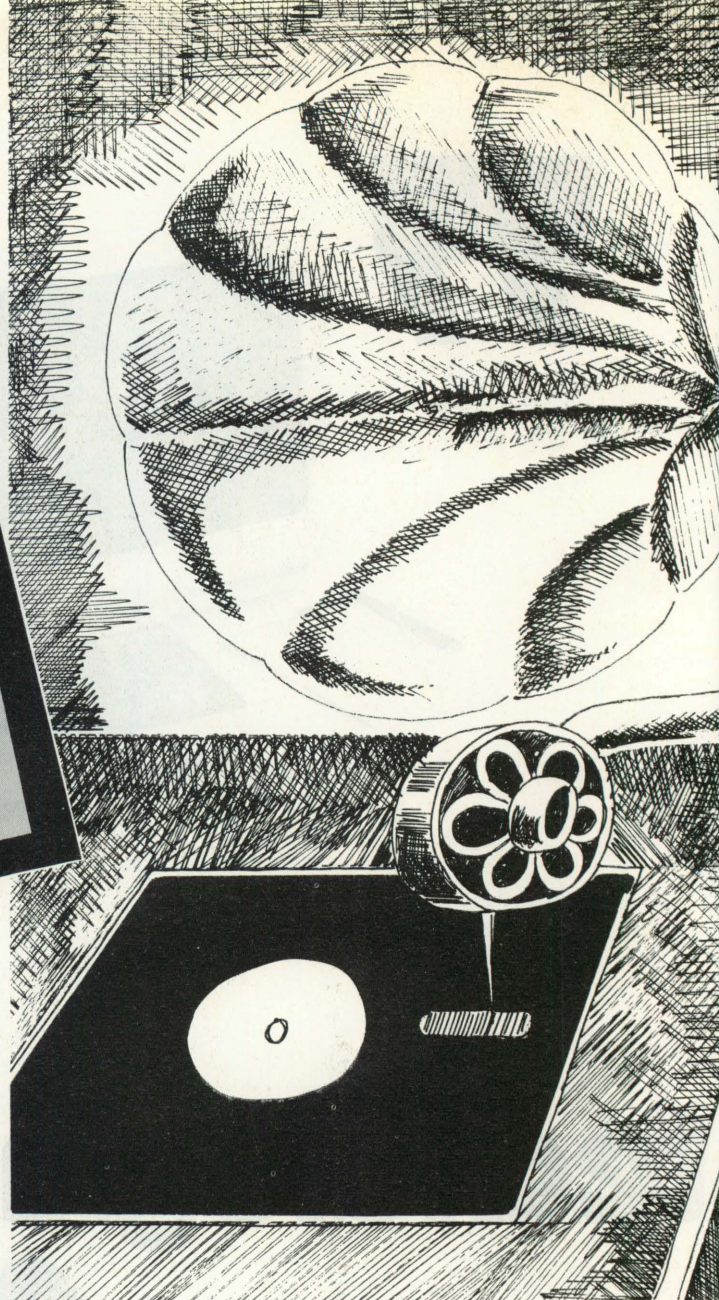
Scopo di questo articolo è per l'appunto di dare all'utente di qualsiasi macchina, dal Sinclair all'Apple, la possibilità di decifrare con facilità dallo spartito musicale gli elementi indispensabili per tradurre il brano contenuto. Sarà poi compito del programmatore trasformare le note e i rispettivi valori nel codice e formato richiesti dalla macchina.

DO	#/b	RE	#/b	MI	FA	#/b	SOL	#/b	LA	#/b	SI
Ø	1	2	3	4	5	6	7	8	9	1Ø	11

Figura 1 - Successione dei semitoni.

Un po' di teoria

Qualsiasi sistema musicale antico o moderno si basa sull'ottava che, come insegna l'acustica, equivale al raddoppio della frequenza (se una nota è di 220 Hz, la stessa nota all'ottava superiore sarà di 440 Hz); a seconda poi della collocazione storico-geografica del sistema, l'ottava è divisa in un numero variabile di parti non necessariamente uguali tra loro.

Le origini del nostro sistema attuale e tradizionale, risalgono a circa due secoli fa, quando il sommo J. Sebastian Bach pubblicò una raccolta di pezzi intitolata: "Il clavicembalo ben

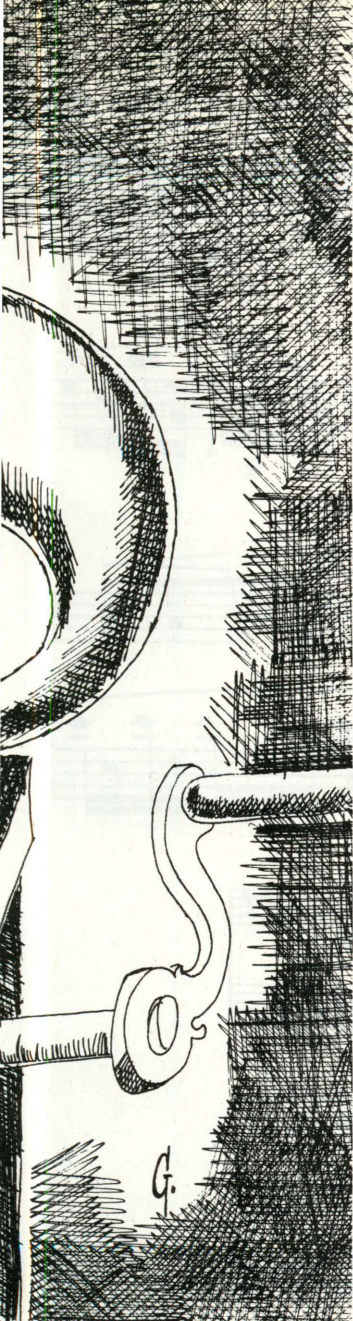
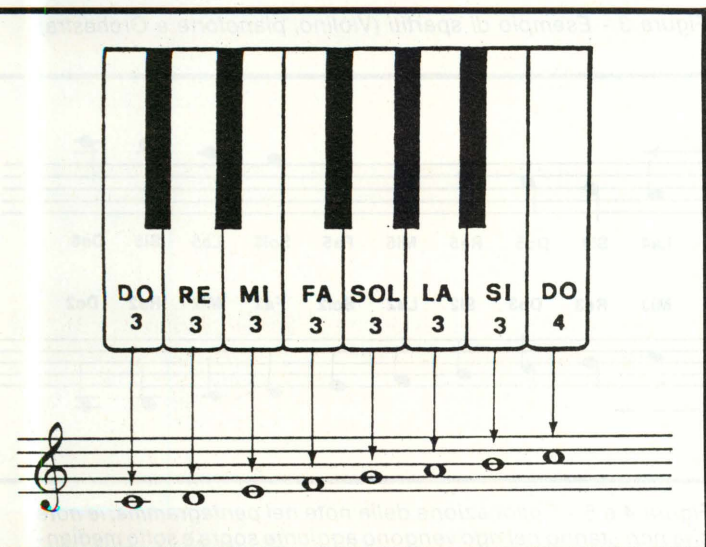
La musica, il software e la vita più dolce

temperato", per la quale lo stesso Bach si fece preparare uno strumento accordato secondo i nuovi criteri. Il nuovo sistema si mostrò subito comodo nella pratica e fu adottato come stadard (*le cose, forse, non sono andate in modo così semplice, né Bach è da considerarsi "l'inventore" della scala temperata, però non sottilizziamo* NdR).
Pur essendo migliore del precedente, anche questo presenta imperfezioni e contraddizioni matematiche che però vengono superate nella pratica correggendo "ad orecchio" gli intervalli imperfetti. È per questo che i dispositivi elettronici di accordatura non danno risultati troppo buoni.

Figura 2 - Corrispondenza tra le note e i tasti di una ottava. A seconda dello strumento la tastiera può essere formata da varie ottave; per esempio il pianoforte ne ha 7, l'organo 5 ecc. Si noti la succesione dei tasti neri (2+3) che permette di identificare univocamente ogni nota.

Come è abbastanza noto, nella scala temperata, cromatica, l'ottava è divisa in dodici parti uguali, chiamate semitoni, cui corrispondono altrettante note musicali.
Tradotto in termini matematci, ogni semitono vale la radice dodicesima di due ($\sqrt[12]{2}$ = 1,05946). Ciò deriva dal fatto che in musica, per comodità (*ma anche - c'è chi sostiene - perché ciò corrisponde ad una legge fisiologica per cui l'orecchio umano è sensibile ai rapporti e non alle differenze di frequenza* NdR), ogni nota è considerata *in rapporto* alle altre note.Tale rapporto si esprime con una frazione o, più comodamente, con un numero decimale e si può ovviamente calcolare dividendo la frequenza delle due note interessate. Volendo invece conoscere la frequenza di una nota, basta moltiplicare quella della nota base per il rapporto. Infine è possibile conoscere il rapporto anche moltiplicando quello relativo ad un semitono per il numero delle note (*toni e semitoni della scala dodecafonica* NdR) presenti tra quella base e quella interessata.
Facciamo un esempio chiarificatore.
Si vuol conoscere il rapporto tra un LA (440 Hz) e la seconda nota successiva (SI). Dato che l'intervallo è pari a due semitoni (in mezzo c'è il SI bemolle) si avrà:
$r = (\sqrt[12]{2})^2 = 1,05946^2$
ossia: $r = 1,12246$
E la frequenza del SI, ove la si desideri, è data da: $f_{SI} = f_{LA} \times r$ = 440 x 1,12246 Hz.
La figura 1 mostra la successione dei 12 semitoni. Ricordiamo brevemente che i due simboli # (diesis) e b (bemolle) hanno la funzione, rispettivamente, di alzare e abbassare di mezzo tono la nota, che ogni semitono della scala cromatica, può avere due nomi, a seconda della provenienza: il Mi può essere anche un Fab, un Fa può essere un Mi#, un Do può essere anche un Si#, ecc. Risiede appunto nel fatto che i diesis coincidano coi bemolli, la prerogativa di fondo del sistema ben temperato, dato che in quelli precedenti non esisteva una precisa relazione tra i diesis e bemolli (il Fa# aveva frequenza diversa dal Solb).
In figura 2 è mostrata la corrispondenza tra i semitoni e i tasti di una qualsiasi tastiera.

Come si scrive

La base su cui vengono posti i segni grafici rappresentanti le note e i relativi valori, è un insieme di 5 righe parallele, chiamate pentagramma o rigo.
A seconda dello strumento per il quale la musica è scritta, si possono avere più pentagrammi sovrapposti (vedi figura 3). Per esempio il pianoforte ne usa due, generalmente uno per mano; lo stesso vale anche per tutti gli strumenti a tastiera, fatta eccezione per l'organo classico che ne usa un terzo per la pedaliera. Sia gli strumenti ad arco che quelli a fiato ne usano uno soltanto.
Esistono poi delle partiture per insieme di strumenti che usano un numero variabile di righi (uno per ogni tipo di strumento). Queste vengono usate da chi deve seguire contemporneamente tutti gli strumenti, come il direttore d'orchestra.
Su un singolo rigo è possibile scrivere più voci contemporaneamente; vedremo poi come sia possibile distinguerle.
Alla dimensione verticale del rigo corrisponde l'altezza delle note; a quella orizzontale, la

Vivace assai
M.S.

13.

Allegro, ma non troppo ♩ = 92
ben marcato
a)
con spirito

20.

Violino I
Violino II
Viola
Violoncello
Contrabasso

Figura 3 - Esempio di spartiti (Violino, pianoforte e Orchestra).

successione temporale.
I cerchietti che rappresentano le note, possono essere scritti o tra due linee (nello spazio), o a cavallo di una linea, in modo cioè che la nota venga tagliata in due dalla riga. Come note nello spazio vengono anche considerate quelle rispettivamente "appese" alla prima linea e quelle "appoggiate" sull'ultima (si veda la figura 4).
Finora si è parlato di altezza dei suoni, ora invece parlerò della durata. È importante non confondere durata e altezza, in *quanto* sono elementi differenti e indipendenti.
Per esprimere i valori, si usano delle frazioni. È logico pensare anche che i valori non vengono considerati in assoluto, ma in rapporto con un valore base, che in questo caso è 4/4 o un intero.
Salvo rare eccezioni, l'intero è anche il valore più grande. Prima di continuare, per chi non l'avesse chiaro, ricordo che come valore si intende quanto un suono dura nel tempo.
È importante capire che per distinguere una melodia, non importa tanto che le note siano suonate lente o veloci, ma che vengano rispettati i loro rapporti. Per esempio, supponiamo di avere questa successione di

Figura 4 e 5 - Collocazione delle note nel pentagramma; le note che non stanno nel rigo vengono aggiunte sopra e sotto mediante i tagli addizionali.

La musica, il software e la vita più dolce

Figura 6 - Esempio di disalterazione.

Figura 7 - Valori delle note e delle pause, e loro rapporto ritmico. Si noti che le note con la virgoletta, possono anche essere scritte con una (o più linee) che le lega assieme.

Figura 8 - Esempio di scrittura di più voci in un unico rigo. I numeri indicano l'ordine delle voci.

note: Do3 4/4, Re3 2/4, Mi3 2/4; se la prima dura 5 secondi, le due successive devono durare 2,5 sec.; se la prima dura 10 sec., le due successive dureranno 5 sec.

I valori delle note sono: 4/4, 2/4, 1/4, 1/8, 1/16, 1/32, 1/64. Per ottenere valori intermedi si mette un puntino dopo la nota (.), il che ne aumenta il valore di metà.
Tornando all'altezza delle note, ricordiamo che è possibile scriverle tutte (sono più di 90) su un pentagramma, che al massimo ne può contenere una ventina, anzitutto aggiungendo un certo numero di linee immaginarie, sia sopra che sotto il rigo, che vengono però tracciate solo in corrispondenza delle note (vedi figura 5).
Si osservi ora la figura 3, quel segno che è posto all'inizio di ogni rigo ( ), si chiama "chiave" ed ha la funzione di stabilire la posizione delle note. Cambiando chiave, infatti, se ne varia l'interpretazione sul pentagramma.
In musica esistono sette chiavi, ma ci basta conoscere quella di sol (o di violino) e quella di fa (o di basso), che sono le più usate. Nella figura 5 è mostrata la collocazione delle note nel rigo musicale. Si noti come le note si succedono ordinatamente dal basso all'alto. Riconoscendo, per ogni chiave, la posizione anche di una sola nota, è possibile decifrare tutte le altre. Alcune delle note più alte della chiave di basso coincidono con delle note della chiave di violino.
Le alterazioni (diesis e bemolle) si scrivono facendo precedere la nota con i simboli # e b.
Può accadere che alcune note di un brano debbano essere sempre alterate in tutta l'esecuzione (per esempio tutti i Fa): anziché annotare l'alterazione di volta in volta basterà inserire l'annotazione all'inizio di ogni rigo in corrispondenza della nota da alterare. Nel tradurre in frequenza le note di un brano si faccia attenzione a questo particolare, per evitare stonature.
Se poi nella composizione per una qualsiasi ragione, occorresse riportare una delle note alterate in chiave alla posizione naturale, le si farà precedere dal segno " " (bequadro) (l'esempio è in figura 6).
All'interno di uno spartito è possibile incontrare note alterate occasionalmente: l'alterazione riguarda non solo la nota specificata, ma tutte le eventuali stesse note presenti di seguito in quella battuta, ovvero lo spazio del pentagramma delimitato da due sbarrette verticali.
Per diversificare il valore delle note, si modifica il "cerchietto" scrivendolo pieno o vuoto, con o senza la gambetta. Si veda la figura 7 che riporta la corrispondenza tra i segni grafici e i valori. Nella stessa figura sono mostrati anche i simboli delle pause necessarie ad ottenere momenti di silenzio.
I valori delle note sono regolati dalla frazione che si trova all'inizio di ogni brano subito dopo la chiave: essa indica quanto tempo deve essere contenuto in ogni battuta. Supponiamo che la frazione in chiave sia 3/4: in ogni battuta potrà esserci un numero qualsiasi di note o pause, ma la somma dei loro valori deve essere 3/4. Una battuta, per fare un esempio, di 3/4 può essere composta da tre note di un quarto o da due note ciascuna di un quarto con due di un ottavo.
Ovviamente quanto detto si riferisce ad ogni *singola voce*.
L'ultimo concetto importante da chiarire è quello di Polifonia. Si ha la polifonia quando sono presenti più voci (almeno due),

La musica, il software e la vita più dolce

ed ognuna esegue una parte diversa procedendo parallelamente e in maniera sincronizzata, producendo effetti armonici.
La scrittura delle voci può avvenire o su più pentagrammi paralleli, o su uno stesso pentagramma nella seguente forma:

- Le voci generalmente sono ordinate dall'alto al basso in modo che la voce più acuta sia quella più in alto.
- Nel caso che due voci si debbano incrociare, è possibile distinguerle osservando in che direzione è rivolta la sbarretta di valore: se è verso l'alto, la nota apparterrà alla voce più alta e viceversa.

La figura 8 mostra esempi di quanto detto.
C'e poi da dire che purtroppo la musica, pur caratterizzata da regole, presenta non poche eccezioni.
Non è il caso di scoraggiarsi in quanto si tratta di anomalie facilmente riconoscibili.

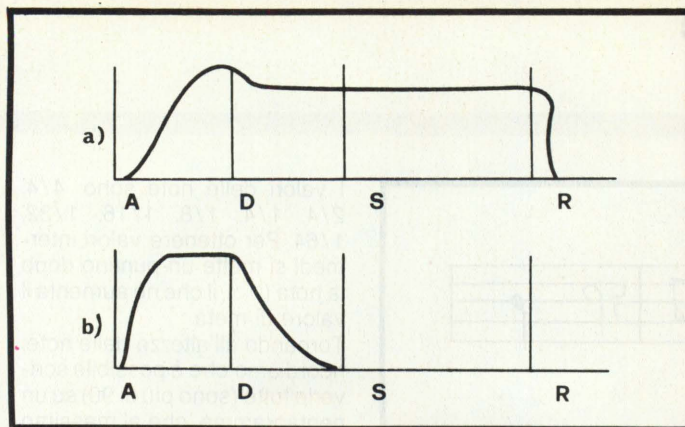



Figura 9 - a) A.D.S.R. tipico di una canna d'organo; A: l'aria viene introdotta nella canna, il suono impiega del tempo a raggiungere la sua massima intensità. D: il flusso d'aria tende a stabilizzarsi dopo che è entrato con forza nella canna. S: ora il flusso è regolare e tale rimarrà fino alla fine della nota. R: le vibrazioni della canna, che non è più alimentata d'aria, si smorzano rapidamente.
b) A.D.S.R. tipico del pianoforte; A: premendo un tasto la corda viene percossa, si ha perciò un tempo di attacco veloce. D: anche tenendo il tasto premuto le vibrazioni della corda si smorzeranno perché la corda non è più alimentata da energia.
Si può dedurre che non tutti i suoni hanno l'inviluppo composto dalle 4 fasi.

A titolo di dimostrazione, ne cito un paio tra le più frequenti:

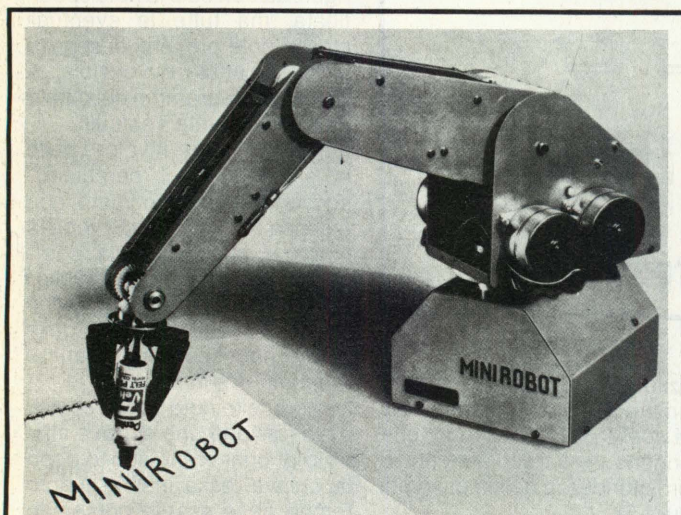
- Si è detto che la somma delle note e pause di ogni battuta deve essere rigorosamente uguale al tempo in chiave: da questa regola possono essere escluse la prima e l'ultima battuta del brano che potranno avere un tempo minore.
- Entro un brano si possono incontrare note scritte più piccole delle altre: si chiamano fioriture o abbellimenti: non si devono contare per stabilire il tempo della battuta e si possono omettere.

In uno spartito sono presenti anche simboli grafici per indicare la dinamica, l'espressione, la tecnica strumentale da usarsi, ecc. Per esempio un puntino o una lineetta sopra la nota indica come il tasto debba essere premuto (violentemente o dolcemente); le lettere minuscole "p, pp, f, ff" indicano la dinamica delle note (volume); delle linee curve tracciate sopra più note, indicano che devono essere suonate molto legate una all'altra; indicazioni come "allegro, andante, adagio, ecc." poste all'inizio del brano, designano la velocità di esecuzione. Evidentemente di tutte queste annotazioni non se ne terrà conto nella trascrizione, non essendo essenziali.

Un po' di fisica acustica

Un sintetizzatore di suoni viene impiegato generalmente in due settori ben distinti:

- cercando di riprodurre il più fedelmente possibile, il suono di strumenti tradizionali;
- creando suoni ed effetti nuovi e comunque non riproducibili con gli strumenti musicali "classici".

Qualsiasi indirizzo si voglia seguire è bene aver chiari gli elementi che caratterizzano un suono.
Ogni suono è composto da tre elementi fondamentali:
ALTEZZA: un suono può essere più grave o più acuto. Questa componente è determinata dalla frequenza dell'onda e per essere percepita dall'orecchio umano deve essere inferiore alla soglia massima di percezione (<20 KHz).
INTENSITA': un suono può essere più forte o più debole; dipende dall'ampiezza dell'onda.
TIMBRO: due suoni di frequenza e intensità eguale possono risultare diversi all'ascolto; ciò dipende dal contenuto armonico dell'onda che ne varia la forma.
Per imitare un suono non basta però fissare dei valori per questi tre parametri, in quanto essi sono soggetti a delle variazioni più o meno consistenti che si verificano nel tempo. Il più soggetto a variare dei tre, è quello dell'ampiezza, specialmente all'inizio e alla fine di un suono.
Si possono distinguere quattro fasi denominate: attacco, decadimento, sostegno e rilascio (A.D.S.R.).
L'attacco è il tempo che il suono impiega a raggiungere la sua massima intensità; raggiunto questo livello, il suono tenderà a cedere per poi stabilizzarsi per tutta la sua durata ad un livello costante, chiamato sostegno. La fase che sta tra quella di attacco e quella di sostegno, si chiama decadimento. Quando poi il corpo vibrante non è più alimentato da energia, il suono andrà sfumandosi fino alla sua estinzione producendo la fase di rilascio. Si veda la figura 9.
Anche il contenuto *armonico* dell'onda può variare in concomitanza con le A.D.S.R. dell'ampiezza, oppure anche in modo indipendente.
Il suono può essere anche modulato sia in frequenza (Vibrato), sia in ampiezza (Tremolo). ■

ECO 1

L'HARDWARE

Direttamente derivato dalla esperienza VDS nei sofisticati sistemi di informatica grafica, l'hardware dei sistemi ECO1 si distingue per la sua impostazione decisamente razionale.

Semplice da installare (una sola scheda perfettamente accessibile), è ancora più semplice da assistere: programmi autodiagnostici residenti sono infatti in grado di segnalare all'operatore qualsiasi possibile avaria sulla macchina.

Dati tecnici: Z80A, 64KB RAM, fino a 8 KB EPROM, 1 o 2 linee seriali RS232, 1 uscita parallela per stampante, video 12" alta risoluzione 24x80, tastiera separata, 2 floppy (2,4 MB) oppure disco fisso fino a 10MB + floppy 1,2 MB.

IL SOFTWARE

Logica conseguenza di una impostazione razionale, i sistemi ECO1 adottano come software di base il più classico degli standard: sistema operativo CP/M® 2.2 originale della DIGITAL RESEARCH, integrato da tutti i linguaggi e compilatori disponibili dalla MICROSOFT; il tutto con regolare licenza d'uso.
Per il software applicativo, programmi originali appositamente realizzati per i sistemi ECO1, molto curati, attentamente collaudati e ben documentati.
Disponibili i sorgenti per le necessarie personalizzazioni.

VIDEO DISPLAY SYSTEMS

LA DISTRIBUZIONE

I sistemi di elaborazione dati ECO1 sono progettati e prodotti da

VIDEO DISPLAY SYSTEMS

V.D.S. VIDEO DISPLAY SYSTEMS - Via G. del Pian dei Carpini 1 - 50127 FIRENZE / Tel. (055)-4378831/4378832 - Telex 573090 MYSA
La distribuzione per l'Italia dei sistemi ECO1 e la assistenza tecnica (hardware e software) è curata da

DEDO SISTEMI - Piazza Indipendenza 13 - 50129 FIRENZE / Tel. (055)-474467/486265 - Telex 574500 DEDOSIS

dBase II un data base programmabile

di **R. Dadda**

Parte prima

Chiunque abbia provato a scrivere un programma in BASIC od in un altro linguaggio analogo, per la gestione dati, ad esempio un indirizzario personale, un archivio di dischi o libri od una gestione del proprio conto in banca, avrà certamente notato che esistono tutta una serie di operazioni per lo più comuni a programmi di questo tipo, estremamente onerose in termini di tempo di programmazione e di correzione dei programmi e molto delicate dal punto di vista della loro manutenzione: c'è il grosso rischio che il programma, se non è strutturato e documentato più che bene risulti incomprensibile dopo pochi mesi allo stesso programmatore che lo ha scritto.

Le operazioni cui mi riferisco sono tipicamente quelle di ingresso ed uscita di dati e quelle di gestione dei file, del loro accesso, del loro ordinamento e delle operazioni di confronto ed interazione tra più file: i programmatori passano molte ore a scrivere *PRINT* di maschere di ingresso su video, controlli di conseguenza sui dati immessi (il programma non deve accettare date del tipo 99/88/83!) e ad accedere ai file per cercare record che sottostiano a particolari caratteristiche (esempio: tutti i clienti che abbiano un sospeso di cassa anteriore ad una certa data).

Era ovvio che le case produttrici di software si sarebbero messe a lavorare per risolvere almeno in parte questo problema creando strumenti di lavoro che potessero fare risparmiare tempo, e di conseguenza denaro, nella fase di programmazione e che potessero generare prodotti programma più manutenibili ed affidabili: il frutto di questa attività di ricerca e sviluppo iniziata sui calcolatori di grandi dimensioni è oggi disponibile anche agli utilizzatori di personal computer sotto forma di pacchetti di programmi che permettono anche a chi non è esperto programmatore la stesura di programmi anche molto complessi in tempi molto brevi: il programma di cui parlo in questi articoli è un data base molto sofisticato che permette la stesura di programmi di im-

CARATTERISTICHE RICHIESTE AL SISTEMA
Sistemi con CPU 8080, 8085 o Z80
48 Kbyte di RAM disponibile
Sistema operativo CP/M versioni 1.4 o 2.X, CDOS o CROMIX
Uno o più sistemi di memoria di massa, tipicamente dischetti
Opzionali: cursore indirizzabile e stampante

CARATTERISTICHE OPERATIVE	
Massimo numero di record per data base	65535
Massima lunghezza del record	1000
Massimo numero campi per record	32
Massima lunghezza dei campi	254
Massima lunghezza del campo indice	100

Tabella 1 - Caratteristiche minime per far girare dBase II e caratteristiche operative dal data base stesso.

```
XDIR B:
Extended Directory version 3.5
           .          2k
AUTOST   .COM      2k
DBASE    .COM     18k
DBASEAPP.OVR      4k
DBASEBRO.OVR      2k
DBASEJOI.OVR      2k
DBASEMAI.OVR      8k
DBASEMOD.OVR      4k
DBASEMSC.OVR      4k
DBASEMSG.COM      8k
DBASERPG.OVR      6k
DBASESRT.OVR      2k
DBASETTL.OVR      2k
DBASEUPD.OVR      2k
IND01    .DBF     14k
PIP      .COM      8k

Disk B:  2K blocks
Size=   92K, 16 Files, Used=    90K, Space=    2K
A>
```

Figura 1 - Directory del disco contenente i programmi originali del dBase II, cui è stato aggiunto per comodità il PIP e il file di dati da back up.

Figura 2 - Directory del disco contenente i programmi scritti nel "linguaggio" del dBase II e i file di dati.

```
ERA
B>A:XDIR B:
Extended Directory version 3.5

IND01    .DBF   14k
IND01    .MEM    4k
IND01    .NDX    8k
INDAPP   .CMD    2k
INDEDTIT.CMD     2k
MENU     .CMD    4k
STAMBEEP.CMD     2k
STAMETT  .CMD    4k
STAMIND  .CMD    2k
STAMTEL  .CMD    2k

Disk B:  2K blocks
Size=   92K, 10 Files, Used=    46K, Space=    46K
B>
```

dBase II
un data base programmabile

```
DISPLAY STRUCTURE
STRUCTURE FOR FILE:   B:IND01.DBF
NUMBER OF RECORDS:    00105
DATE OF LAST UPDATE: 04/04/83
PRIMARY USE DATABASE
FLD        NAME      TYPE WIDTH   DEC
001        NOME        C    025
002        INDIRIZZO   C    020
003        LOCAL       C    015
004        TELEFONO    C    010
005        PREFISSO    C    005
006        VARIE       C    040
007        CODICE      C    003
** TOTAL **               00119
```

Figura 3 - Struttura dei file IND01.DBF.

Figura 4 - Listato del main program della gestione indirizzi.

```
TYP TYPE MENU.CMD
NOTE - FILE DI COMANDI DBASEII - GESTIONE INDIRIZIARIO
NOTE -      roberto dadda         gennaio 1983
*
*
*  - elimina i messaggi di sistema sullo schermo
SET TALK OFF
*  - scelta del file da utilizzare
USE B:IND01
*  - stampa del menu' sullo schermo
DO WHILE T
* DO WHILE T  significa continua per sempre (T e' sempre .TRUE.)
*    Questa DO LOOP si ferma solo con il  CANCEL alla scelta = 0
ERASE
  ?
  ? '     ********************************'
  ? '     * INDIRIZIARIO DI ROBERTO DADDA *'
  ? '     ********************************'
  ?
  ?
  ? '      0 - FINE'
  ? '      1 - AGGIUNTA INDIRIZZI NUOVI'
  ? '      2 - CORREZIONE SU MATRICE'
  ? '      3 - CORREZIONE SINGOLO RECORD O CANCELLAZIONE'
  ? '      4 - STAMPA INDIRIZIARIO GENERALE'
  ? '      5 - STAMPA RUBRICA TELEFONICA'
  ?
  ?
  ? ' SCEGLI UNA DELLE OPZIONI '
  ?
  ?
  *  - attesa della scelta dell'utente'
  WAIT TO SCELTA
  DO CASE

    *  - fine
    CASE SCELTA='0'
      SET TALK ON
      *  - ritorno al sistema
      ERASE
      ? CHR(7),'BYE ...'
      QUIT

    *  - aggiunta indirizzi nuovi
    CASE SCELTA ='1'
      *  - richiama programma append e sort
      DO INDAPP

    *  - correzione su matrice
    CASE SCELTA='2'
      *  - comando di correzione su video
      BROWSE

    *  - correzione singolo record o cancellazione
    CASE SCELTA='3'
      *  - chiama programma correzione record
      DO INDEDIT

    *  - stampa indiriziario generale
    CASE SCELTA='4'
      *  - chiama il programma di stampa
      DO STAMIND

    *  - stampa rubrica telefonica
    CASE SCELTA='5'
      *  - chiama il programma di stampa rubrica
      DO STAMTEL

  * - comando non valido
    OTHERWISE
      ? 'Comando non valido     RIPROVA !'

  ENDCASE

ENDDO
RETURN
```

```
B>TYØPE B:INDAPP.CMD
NOTE - FILE DI COMANDI PER AGGIUNTA INDIRIZZI E SORT
*
ERASE
?
? '        Immissione nuovi indirizzi'
?
? '     ********************************'
? '     * quando hai finito premi RETURN *'
? '     *      al posto del nome         *'
? '     ********************************'
?
? '  PREMI QUALSIASI TASTO PER CONTINUARE'
WAIT
*
* - comando di aggiunta record
APPEND
SET TALK ON
ERASE
? '     ***************************'
? '     * STO FACENDO IL    S O R T *'
? '     ***************************'
?
* - sort e backup
SORT ON NOME TO A:IND01
* - usa il nuovo file sortito
ERASE
? '     ***********************************'
? '     * STO FACENDO LA COPIA DI SICUREZZA *'
? '     ***********************************'
?
USE A:IND01
* - copialo sul disco di lavoro
COPY TO B:IND01
* - rimetti in uso il file di lavoro lasciando
* - l'altro come backup
USE B:IND01
SET TALK OFF
ENDDO
```

Figura 5 - Listato del programma di ingresso nuovi indirizzi, di messa in ordine alfabetico e di esecuzione della copia di sicurezza.

missione, elaborazione e stampa di dati anche all'utente che non vuole sapere nulla di programmazione, ma che vuole limitarsi a dare istruzioni in inglese per dire alla macchina cosa deve fare!

Il dBase II

Il programma in esame è stato scritto dalla californiana Ashton-Tate e funziona, come indicato nella tabella 1 sotto CP/M su quasi tutti i sistemi che possono utilizzare questo sistema operativo ivi compreso Apple II con la scheda Z80.
Il programma è scritto in assembler 8080 che come noto è un subset dell'assembler Z80 ed è contenuto in un primo dischetto di distribuzione il cui directory è riportato in figura 1: come potete vedere si tratta di ben 62 Kbyte di programmi assembler che sono una bella quantità di lavoro se si considera che l'intero interprete MBASIC per la stessa macchina è lungo 24 Kbyte e che tutto supercalc è lungo 44 Kbyte! (la versione da me utilizzata del programma è quella Osborne I commercializzata direttamente da IRET ed i listati di prova sono stati ottenuti con una macchina ancora nella versione con dischetti a singola faccia e singola densità).

Il file chiamato all'inizio delle operazioni è il *DBASE.COM*, mentre quelli marchiati con l'estensione *.OVR* contengono moduli di servizio che vengono automaticamente chiamati dal programma stesso al bisogno.
Il programma viene venduto su dischetto copiabile, ovviamente, come quasi sempre sotto CP/M, fidando che l'utente utilizzi queste possibilità per la sola funzione di back up, e corredato da un ampio e dettagliatissimo manuale operativo di circa 400 pagine: si tratta di un malloppone grande praticamente come tutto il manuale Osborne!
In questo primo articolo presenterò alcune delle possibilità operative di questo pacchetto utilizzando un programma che uso regolarmente per tenere la mia agenda di indirizzi: fine degli articoli è il dare un'idea delle possibilità offerte dal programma e non darne una esauriente descrizione per la quale rimando gli interessati al manuale operativo dal momento che per descrivere a fondo le caratteristiche del programma non basterebbe una annata di rivista!
La seconda parte della tabella 1 presenta alcune delle caratteristiche operative del data base e chi ha già avuto a che fare con programmi di questo tipo noterà come le caratteristiche di dBase II non sfigurano anche

se paragonate a quelle di data base che girano anche su macchine ben più grandi dei personal.

Un semplice esempio

La figura 2 riporta il directory del dischetto che contiene il "programma" scritto nel linguaggio del dBase II che io utilizzo regolarmente per gestire la mia agenda di indirizzi: i file marcati con la estensione .CMD contengono una serie di comandi del database e possono essere richiamati con la funzione DO nome del file che provoca la ricerca del file sul disco e la esecuzione delle istruzioni contenute riga per riga; si noti che le stesse isruzioni possono essere eseguite da tastiera una per una con gli stessi identici risultati, si tratta dunque di qualche cosa di simile ai linguaggi tipo *EXEC* o al *SUBMIT* del CP/M.

Il file *IND01.DBF* contiene il database vero e proprio ed è stato creato con la istruzione *CREATE* mediante la semplice immissione da tastiera, guidata da dBase II, della definizione dei campi riportata nella figura 3: di ogni campo viene fornito il nome, la natura e la lunghezza accompagnata, per quelli numerici, dal numero di decimali presenti; la descrizione data nella figura è stata ottenuta semplicemente con la istruzione *DISPLAY STRUCTURE* che permette di ottenere una completa documentazione del file di database.

Il programma di gestione della banca dati viene lanciato con la istruzione *DO MENU* che provoca la esecuzione dei comandi contenuti nel file *MENU.CMD* che è riportato nella figura 4.

Il file contiene la scelta del file di dati (USE...) da utilizzare e la stampa di un menu di possibili operazioni; dopo la stampa su video di menu il programma si ferma fino alla pressione di un tasto alla istruzione *WAIT* ed il valore del tasto premuto viene assegnato alla variabile *SCELTA* che viene usata nella successiva struttura del tipo *CASE* per la messa in esecuzione vuoi di altri file di comandi, vuoi di comandi di sistema.

La presentazione del menu e la attivazione dei comandi è posta in una loop infinita che si interrompe solo alla immissione di uno 0 con il *QUIT* preceduto dalla pulizia del video (*ERASE*) e dal suono del *BEEP* del terminale (*? (= PRINT) CHR$ (7))*.

La pressione del tasto 2 provoca la messa in esecuzione del programma contenuto nel file *INDAPP.CMD* che serve alla introduzione dei dati e che è riportato nella figura 5.

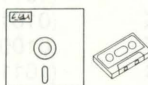
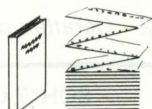
Il file *INDAPP.CMD* inizia con la pulizia dello schermo e con la stampa sul video di alcune istruzioni, seguita dalla istruzione *WAIT* senza specifica di nome di variabile che ferma il programma fintanto che un tasto qualsiasi non sia stato premuto; la successiva istruzione *APPEND* provoca la visualizzazione sul video di una maschera di input/output generata in forma completamente automatica da dBase II per la immissione dei dati.

Alla fine della immissione, segnalata dalla pressione del *RETURN* in luogo del primo campo, il programma prosegue con un *SET TALK ON* che, contrapposto al *SET TALK OFF* ordinato all'inizio del *MENU.CMD*, provoca la stampa dei messaggi di sistema in modo che l'utente sia avvisato di cosa la macchina sta facendo nelle fasi di sort e back up in modo automatico.

La messa in ordine alfabetico sul file di *BACK UP* contenuto nel disco A: è fatta in modo assolutamente automatico dopo la emissione del comando di *SORT* dove viene specificato rispetto a quale campo sortire (*ON NOMNE*) e dove mettere il risultato del sort (*TO A:IND01*). Una volta sortito il file sul disco A:, quest'ultimo viene messo in uso e poi ricopiato (*COPY*) sul file originale sul disco B:; il file sul disco B: viene infine messo in uso e quello sul disco A: lasciato come back up.

ENDDO segnala infine la fine del file di comandi e provoca il ritorno al file chiamante con la stampa di nuovo del menu.

Come avete visto con pochi minuti di lavoro si è ottenuto un programma di immissione dati che altrimenti avrebbe necessitato di molte ore di programmazione, ma il meglio ha ancora da venire.

(continua)

Trasmissione dati: un protocollo di comunicazione

di **F. Merlo**

Parte prima

Finalità

Dopo aver presentato su **Bit** n. 24 (gennaio '82) un'introduzione sui problemi relativi alla trasmissione dati, riteniamo utile presentare al lettore un protocollo di comunicazione per accoppiatori acustici. Lo scopo che ci prefiggiamo è di fornire al lettore un software in grado di gestire un collegamento punto a punto tra due sistemi, per mezzo di una linea commutata e di accoppiatori acustici. Si ritiene opportuno, prima di procedere alla descrizione del programma, riproporre brevemente alcune definizioni già fornite nell'articolo citato.

- Accoppiatori acustici (acoustic coupler):
Sono apparati che consentono, collegati ad un normale apparecchio telefonico, di spedire e ricevere dati sulla linea telefonica pubblica, usando la normale rete Sip, senza tracciato prefissato. Per riconoscere gli stati logici 0 (space) ed 1 (mark), si opera su due frequenze diverse, scelte nella banda "fonica" tra 300 e 3200 Hz, sufficientemente differenziate, per evitare di confondere i caratteri.

- Linea commutata:
Il collegamento tramite rete commutata (public switched line) utilizza una linea fisica che passa attraverso le centrali di commutazione esistenti (a livello urbano, settoriale, distrettuale, ecc.) e può seguire, di volta in volta, un tracciato fisico diverso.
La trasmissione dei dati si effettua o con un modem commutatofonia/dati o, come nel nostro caso, con un accoppiatore acustico collegato alla normale cornetta del telefono.

- Affidabilità:
Per affidabilità intendiamo, nel caso della trasmissione dati, la capacità del sistema di inviare e ricevere dati corretti, utilizzando, eventualmente, delle tecniche di verifica e correzione che comportino il minor uso possibile della rete telefonica.

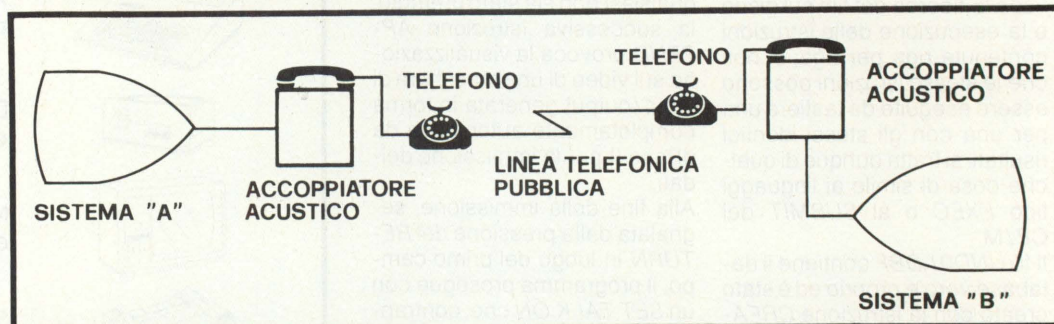



Figura 1 - Schema di collegamento P.T.P. su rete commutata.

ENQ	(0101000)	ENQUIRY
ACK	(0110000)	ACKNOWLEDGE
SOH	(0001000)	START OF HEADER
STX	(0010000)	START OF TEXT
ETX	(0011000)	END OF TEXT
NAK	(0101001)	NEGATIVE ACKNOWLEDGE
EOT	(0100000)	END OF TRANSMISSION
ESC	(1011001)	ESCAPE

Figura 2 - Caratteri di controllo utilizzati dal protocollo.

SOH/7 CRTS DI IDENTIFICAZIONE/
I CRT PO/ I CRT PV/ ETX
Blocco di identificazione

STX/ 7CRTS DATI/ I CRT PO/ I CRT PV/ ETX
Blocco dati

ESC/ COMANDO/ PO/PV/ ETX
Blocco comandi

Figura 3 - Blocchi di trasmissione per i vari formati.

	1 BIT DI PARITA'	
0001110	1 bit di parità	(corretto)
0101110	1 bit di parità	(errore rilevato) (a)
1101110	1 bit di parità	(errore non rilevato) (b)

Figura 4 - Rilevazione di errori con il solo bit di PO.

Operando su rete commutata dobbiamo tener conto dei disturbi indotti dai sistemi di commutazione, dalla diafonia e dalle microinterruzioni, che fastidiosi durante una normale conversazione telefonica, possono comportare notevoli errori nella trasmissione dati.
Bisogna ricordare, inoltre, che alcune tratte della rete telefonica italiana sono considerate "secondarie" e servite, ancora, con centrali di commutazione di concezione tecnologica superata.
Il "tasso di errore", cioè il rapporto bit errati/bit trasmessi può essere considerato, su queste tratte, pari a n ★ 10^{-3}, quindi, ogni 1000 bit trasmessi possono esservene 'n' errati.
È compito del protocollo di comunicazione, cioè del software di gestione del collegamento, mettere a disposizione dell'utente dei mezzi di individuazione e correzione errori che consentano di ridurre il tasso effettivo di errore.

- Protocolli di comunicazione:
Governano lo scambio di messaggi tra i diversi elementi di un sistema, verificando la corretta

ricezione dei dati trasmessi; in caso di errore o richiedono la trasmissione o procedono, se possibile, alla correzione automatica.
Provvedono anche a sincronizzare trasmettitore e ricevitore, in modo che il colloquio possa svolgersi regolarmente.
- *Collegamento punto a punto (point to point):*
È il collegamento di tipo più semplice e può essere usato anche per linee non commutate (riservate).
Ogni linea collega soltanto due utenti, al contrario del collegamento multipunto, ed il protocollo di comunicazione è notevolmente semplificato (figura 1).

Il protocollo proposto

Il protocollo di comunicazione che proponiamo opera con un codice USASCII-CCITT 5-ISO 7 bit CODE.
Per la gestione del colloquio vengono usati 8 caratteri (figura 2), detti caratteri di controllo.
La correzione degli errori avviene per mezzo di una matrice 7x7 bit, che obbliga, per una gestione ottimale, alla trasmissione di blocchi fissi di 7 caratteri, ma che consente, in cambio, un'ottima rilevazione degli errori e la correzione di quasi tutti gli errori di ordine dispari contenuti in ogni singolo blocco. Inoltre, operando con blocchi corti, si hanno meno possibilità di errore.

```
                      PO
     0 1 1 0 1 0 1    0
     1 0 0 1 1 0 0    1
     0 0 1 0 1 0 0    0
     1 0 1 1 0 1 0    0
     0 0 0 1 1 1 0    1
     0 1 1 1 1 0 1    1
     1 1 1 1 0 0 0    0

PV   1 1 1 1 1 0 0
```

Figura 5a - Matrice 7x7 corretta.

```
                      PO
     1 1 1 0 1 0 1    0  →
     1 0 0 1 1 0 1    1  →
     0 0 1 0 1 0 0    0
     1 0 1 1 0 1 0    0
     0 0 0 1 1 1 0    1
     0 1 1 1 1 0 1    1
     1 1 1 1 0 0 0    0

PV   1 1 1 1 1 0 0
     ↑           ↑
```

Figura 5b - Matrice 7x7 rilevazione di un errore di secondo ordine.

```
                      PO
     0 1 1 0 1 0 1    0
     1 0 1 1 0 0 0    1  →
     0 0 1 0 1 0 0    0
     1 0 1 1 0 1 0    0
     0 0 1 1 0 1 0    1  →
     0 1 1 1 1 0 1    1
     1 1 1 1 0 0 0    0

PV   1 1 1 1 1 0 0
             ↑ ↑
```

Figura 5c - Matrice 7x7 errore di quarto ordine non rilevabile.

Figura 5 - Rilevazione errori tramite matrice.
N.B.: dall'esempio è evidente che gli errori di quarto ordine non sono rilevabili quando formano i vertici di un rettangolo.

In un paragrafo successivo verrà approfondito il funzionamento delle matrice proposta.
Il protocollo consente, per mezzo di un uso appropriato dei caratteri di controllo, di:

— Identificare il trasmittente ed il ricevente;

— identificare la natura dei caratteri trasmessi (dati o comandi);

— trasmettere blocchi a lunghezza fissa (7 caratteri);

— gestire la matrice di controllo degli errori;

— segnalare la corretta ricezione, o invitare a ritrasmettere;

— ritrasmettere il blocco in caso di errore non correggibile.

L'identificazione del trasmittente e del ricevente, effettuata automaticamente, serve ad evitare, operando su rete telefonica pubblica, che un utente non autorizzato acceda a dati riservati. La suddivisione dei dati trasmessi in "dati" veri e propri ed in comandi è prevista per evitare che una configurazione di bit riproducente un comando dia luogo, per esempio durante la trasmissione di un dump, ad effetti indesiderati.

E CHI MI AIUTERA' A FAR CRESCERE IL MIO GIRO D'AFFARI?

GGK

IL PERSONAL COMPUTER IBM. IL TUO PICCOLO GRANDE AMICO.

Quando gli affari aumentano, crescono le soddisfazioni, ma cresce anche la mole di lavoro. Senza una perfetta organizzazione, rischi di rimanere intrappolato.
Ma oggi c'è un amico per te, pronto a darti una mano. È il Personal Computer IBM.
Ti aiuta a snellire e risolvere tutti i problemi quotidiani della tua attività. E non solo quelli. Perchè il Personal Computer IBM può ricevere dati, calcolare, gestire l'archivio, il magazzino, la contabilità e i preventivi. E in pochissimo tempo potrai stampare tutto quello che ti serve.
Vedrai, in poche ore diventerete ottimi amici, perchè ragiona come te. Vuoi metterlo alla prova? Vai da un concessionario IBM per il Personal Computer IBM. Il tuo piccolo grande amico ti sta aspettando.

Il Personal Computer IBM contiene un microprocessore a 16 bit e una memoria di utilizzo che raggiunge i 640 Kbyte. E, grazie ai dischi fissi, la capacità massima di memoria del sistema è di 21 Mbyte in linea. Inoltre, puoi facilmente collegarti con un altro Personal Computer IBM, con elaboratori più potenti e con la rete dei Centri Servizi Elaborazione Dati della IBM.

Sistemi operativi: DOS 1-DOS2-UCSD-CP/M-86. **Supporti per le comunicazioni:** Supporto per Comunicazioni Asincrone - Supporto per Comunicazioni SDLC - Programma di Emulazione 3101 - Programma di Emulazione 3270.
Programmi applicativi: Corso Autodidattico Interattivo - Gestione Aziendale - EasyWriter (dal 20/5 anche in italiano) - Multiplan (dall'8/6 anche in italiano) - VisiCalc.

Trasmissione dati: un protocollo di comunicazione

```
                     PO
     0 1 1 0 1 0 1   0
     1 0 0 1 1 0 0   1
     1 1 1 1 1 0 0   0  →
     1 0 1 1 0 1 0   0
     0 0 0 1 1 1 0   1
     0 1 1 1 1 0 1   1
     1 1 1 1 0 0 0   0

PV   1 1 1 1 1 0 0
       ↑   ↑     ↑
```

Figura 6a - Matrice 7x7 rilevazione di un errore di terzo ordine, con possibilità di correzione.

```
                     PO
     0 1 1 0 1 0 1   0
     1 0 0 1 1 0 0   1
     1 0 1 1 1 0 0   0  →
     1 1 1 1 0 1 0   0  →
     0 0 0 1 1 1 0   1
     0 1 1 1 1 0 1   1
     1 1 1 1 0 0 0   0

PV   1 1 1 1 1 0 0
     ↑   ↑     ↑
```

Figura 6b - Matrice 7x7 rilevazione di un errore di terzo ordine, senza possibilità di correzione.

Figura 6 - Correzione errori tramite matrice.

Utilizzo dei caratteri di controllo

I caratteri di controllo riportati nella figura 2 vengono utilizzati come segue:

ENQ: è il carattere di richiesta identificativo, viene inviato, ad inizio trasmissione, per chiedere al sistema collegato di identificarsi.

ACK: inviato dal ricevente al trasmittente per comunicare la corretta ricezione dell'ultimo blocco trasmesso, ed inviare alla trasmissione del blocco successivo.

SOH: inviato come primo carattere, indica che il blocco di sette caratteri successivi è l'identificativo del sistema.

STX: indica che i sette caratteri successivi sono appartenenti a "dati" veri e propri.

ESC: indica che i sette caratteri successivi sono appartenenti ad un comando.

NAK: inviato dal ricevente al trasmittente per comunicare che l'ultimo blocco trasmesso contiene errori non correggibili, se ne richiede, pertanto, la ritrasmissione.

EXT: indica la fine del blocco appena trasmesso.

EOT: indica la fine del collegamento.

Oltre a questi otto caratteri di controllo, altri due caratteri sono necessari per la costituzione della matrice di rilevazione errori, sono:

PO: parità orizzontale.

PV: parità verticale.

Nella applicazione presentata verrà utilizzato il controllo di parità, quindi il bit di controllo sarà ad uno se il numero di bit a uno nella riga (PO) o nella colonna (PV) è dispari, a zero se è pari.
La PO è gestita via software per consentire una migliore gestione della rilevazione e correzione degli errori.

Nella figura 3 sono riportati i blocchi di trasmissione per l'identificazione, i "dati" ed i comandi.

Figura 7a - Procedura di trasmissione normale.

Figura 7b - Procedura in caso di somma di errori non correggibili.

Matrice di rilevazione errori

Il sistema più semplice di rilevazione errori si basa esclusivamente sulla presenza di un bit di controllo, e consente di rilevare tutti gli errori di ordine dispari (figura 4a), mentre non rileva gli errori di ordine pari (figura 4b), e non consente di correggere automaticamente gli errori riscontrati, ma richiede la ritrasmissione del carattere.
La matrice proposta, al contrario, consente la rilevazione di tutti gli errori di ordine dispari, di tutti quelli di secondo ordine, e di buona parte di quelli di quarto sesto ed ottavo ordine (figura 5a e 5b).
Sono automaticamente correggibili, inoltre, tutti gli errori di ordine dispari presenti nel medesimo carattere, mentre sono solo parzialmente correggibili in caso contrario (figura 6a e 6b).

Procedura di collegamento

Stabilito il collegamento fisico, e posta la cornetta del telefono sull'accoppiatore acustico, il protocollo di comunicazione provvede alla identificazione dei sistemi collegati.
Se l'identificazione è positiva, procederà all'invio del primo blocco, dopo averne calcolato la PO e la PV, che verranno inviati in coda al blocco dati; infine invierà il carattere EXT.
Al ricevimento di un blocco, provvederà alla ricostruzione della matrice, ricalcolerà la PO e la PV, per confrontarle con quelle ricevute.
In caso di verifica positiva, invierà un carattere ACK, e si predisporrà alla ricezione del blocco successivo.
In caso di verifica negativa, procederà, se possibile, alla correzione degli errori, e successivamente invierà il carattere ACK; se la correzione non è possibile, invierà un carattere NAK, e si predisporrà alla ricezione del blocco ritrasmesso.
In trasmissione, il blocco trasmesso resterà disponibile fino alla conferma di ricezione corretta (ACK), per essere ritrasmesso in caso di NAK.
Terminato il colloquio, invierà un carattere di EOT per segnalare la fine trasmissione.
Nella figura 7 (a e b) è riportato uno schema del colloquio appena descritto.
Il protocollo, inoltre, dispone di ulteriori procedure di controllo, che verranno descritte durante l'esame dettagliato del suo funzionamento.

Conclusioni

Questo primo articolo ha presentato, nelle sue caratteristiche essenziali, un protocollo di comunicazione per accoppiatori acustici. Nel prossimo articolo si comincerà a prendere in esame l'analisi del programma, con i relativi schemi a blocchi e flow chart; infine verranno presentati i listings dei programmi, scritti in linguaggio assembler per utilizzatori di MPU Z80 e 6502. ■

ECCO CHI TI AIUTERA' AD ANDARE D'AMORE E D'ACCORDO CON IL TUO NUOVO AMICO.

Il tuo concessionario IBM.
Ti aiuterà a ottenere il massimo dal tuo Personal Computer IBM. Ti garantirà un'assistenza puntuale e un servizio all'altezza del nome IBM, che in tutto il mondo significa efficienza e affidabilità. Per una lunga e proficua amicizia fra te e il tuo Personal Computer IBM.

Aosta
INFORMATIQUE SAS - Av. Du Cons. Des Commis, 16 - 11100 Aosta - Tel. 0165.2242

Bari
PASED SRL - Via Calefati, 134/136 - 70125 Bari - Tel. 080/481488

Belluno
SCP COMPUTER SYSTEM SRL - Via Feltre, 32 - 32100 Belluno - Tel. 0437.70826

Bergamo
NUOVA INFORMATICA SAS - Via Provinciale, 86 - 24021 Albino - Tel. 035.751784
SELTERING SPA - Via Verdi, 31 - 24100 Bergamo - Tel. 035.248256/7/8

Bologna
ABACO INFORMATICA SAS - Via Bernini, 1 - 40138 Bologna - Tel. 051.393274
C.M.B. INFORMATICA SRL - Via Arcoveggio, 74/10 - 40129 Bologna - Tel. 051.323594
PALAZZO DONATO - Via Emilia, 23/A - 46026 Imola - Tel. 0542.29195
SYSDATA ITALIA SPA - Via M. D'Azeglio, 58 40123 Bologna - Tel. 051.330021

Bolzano
BOPAM SAS - Via C. Battisti, 32 - 39100 Bolzano - Tel. 0471.30113

Brescia
FIN-ECO SERVICE SRL - Via G. Rosa, 34 - 25100 Brescia - Tel. 030.59055
MICROSELT SRL - Via Cipro, 33 - 25125 Brescia - Tel. 030.220391
SELTERING SPA - Via Cipro, 33 - 25125 Brescia - Tel. 030.220391

Cagliari
C.D.S. SAS - Via Sonnino, 108 - 09100 Cagliari - Tel. 070.650756

Campobasso
PUBLISISTEMI SRL - Via S. Antonio Abate, 231 - 86100 Campobasso - Tel. 0874.98141

Como
BRUNO SRL - Via Rubini, 5 - 22100 Como - Tel. 031.260538
ZECCA INFORMATICA SPA - Viale Dante, 14 - 22053 Lecco - Tel. 0341.373290

Cosenza
CALIÒ SRL - Via N. Serra, 90 - 87100 Cosenza - Tel. 0984.32807

Cuneo
SISTEMI SRL - Via Giolitti, 26 - 12100 Cuneo - Tel. 0171.55475/6

Firenze
C.C.S. SAS - Viale Repubblica, 298 - 50047 Prato - Tel. 0574.580222
SAL DISTRIBUZIONE SRL - Punto Vendita SESA - Via delle Panche, 65 50100 Firenze - Tel. 055.411635
SESA DISTRIBUZIONE SRL - Via XI Febbraio, 24 B - 50053 Empoli - Tel. 0571.72148

Forlì
HARD & SOFT SYSTEMS SRL - Via Valturio, 43 - 47037 Rimini - Tel. 0541.773343
I.C.O.T. IMPIANTI SRL - Via Codazzi, 10 47100 Forlì - Tel. 0543.723014

Frosinone
SAIU ELETTRONICA SRL - Via Vado del Tufo, 85 - 03100 Frosinone Tel. 0775.83093

Genova
DIFFEL SRL - Via XX Settembre, 31/4 - 16121 Genova - Tel. 010.592431

Lecce
S.V.I.C. SRL - Via V. Emanuele, 121 - 73024 Maglie - Tel. 0836.21604

Lucca
DELPHI SRL - Via Aurelia Sud, 39 - 55049 Viareggio - Tel. 0584.393068

Messina
SICIL FORNITURE SPA - Via Don Blasco, 75 98100 Messina - Tel. 090.2923987

Milano
DATA OPTIMATION SRL - Via Masaccio, 12 20149 Milano - Tel. 02.4987876
DATA PROGRESS SRL - Via V. Emanuele, 44/A - 20059 Vimercate Tel. 039.667423
EDICONSULT SRL - Via Rosmini, 3 - 20052 Monza - Tel. 039.389850
ELEDRA 3S SPA - Viale Elvezia, 18 - 20154 Milano - Tel. 02.349751
HOMIC PERSONAL COMPUTER SRL - Piazza De Angeli, 3 - 20146 Milano - Tel. 02.4988201
HUGNOT LUIGI LUCIANO - Via De Togni, 10 - 20123 Milano - Tel. 02.873190
MICROTECH SRL - Via F.lli Bronzetti, 20 - 20129 Milano - Tel. 02.733609
S.D.I. STUDIO DI INFORMATICA SPA - Via G. Winckelmann, 1 - 20146 Milano - Tel. 02.4223305
SIRIO SHOP SRL - Viale Certosa, 148 - 20156 Milano - Tel. 02.3010051
SOFTEC SRL - Viale Mayno, 10 - 20129 Milano - Tel. 02.7491196
TRANSDATA SRL - Milano Fiori Palazzo E3 Str 1ª - 20094 Assago - Tel. 02.8242460

Modena
DATA SRL - Via B. Peruzzi, 12 - 41012 Carpi - Tel. 059.688090
DATAX SRL - Via Biondo, 6 - 41012 Carpi - Tel. 059.698355

Napoli
POINTER SRL - Via A. De Gasperi, 45 - 80133 Napoli - Tel. 081.312312

Padova
CERVED ENGINEERING SPA - C.so Stati Uniti, 14 - 35100 Padova - Tel. 049.760733

Palermo
SER.COM. ITALIA SRL - Via Sciuti, 180 90144 Palermo - Tel. 091.261041
TESI SRL - Via E. Notarbartolo, 23 - 90141 Palermo - Tel. 091.260549

Pavia
I.T.C. INFORMATICA SRL - Strada Nuova, 86 - 27100 Pavia - Tel. 0382.303201
LOGICA INFORMATICA SRL - Via Montegrappa, 32 - 27029 Vigevano - Tel. 0381.81888

Perugia
PUCCIUFFICIO SNC - Via XX Settembre, 148C - 06100 Perugia Tel. 075.72992

Roma
CERVED SPA - Via Appia Nuova, 696 - 00100 Roma - Tel. 06.7940241
DATAOFFICE SPA - Via Sicilia, 205 - 00187 Roma - Tel. 06.4754568
ELEDRA 3S SPA - Via G. Valmarana, 63 00100 Roma - Tel. 06.8127324
GEDIN SRL - L.go D. De Dominicis, 7 - 00159 Roma - Tel. 06.432183
I.S.E.D. SPA - Via Tiburtina, Km. 12,300 - 00131 Roma - Tel. 06.4125851
JACOROSSI SPA - Via V. Brancati, 64 - 00144 Roma - Tel. 06.54916
SAPES SRL - V.le Tito Livio, 12 - 00136 Roma - Tel. 06.3453536
VALDE ADEL SRL - Piazza S. Anastasia, 3 00186 Roma - Tel. 06.6786648

Salerno
OMNIA SRL - C.so Garibaldi, 47 - 84100 Salerno - Tel. 089.353914

Siena
SILOG SISTEMI LOGICI SRL - Via Sicilia, 5 - Belverde - 53100 Siena - Tel. 0577.54085

Terni
DPS SRL - Via Pacinotti, 6 - 05100 Terni - Tel. 0744.58247

Torino
DIVERSIFICATE VENCO SRL - C.so Matteotti, 32A - 10121 Torino - Tel. 011.545525
PROGRAMMA SPA - Corso Svizzera, 185 10149 Torino - Tel. 011.746421
SISTEMI SPA - C.so Peschiera, 240 - 10139 Torino - Tel. 011.3358676
SOFTEC SRL - C.so San Maurizio, 79 - 10124 Torino - Tel. 011.8396444

Trento
SIGE SNC - COMPUTER SHOP - Via Prato, 22 - 38100 Trento - Tel. 0461.25154

Treviso
EDS SRL - Via S. Pio X, 154 - 31033 Castelfranco Veneto - Tel. 0423.490178
INFORMATICA TRE SRL - Viale della Repubblica, 19 - 31100 Treviso - Tel. 0422.65993

Trieste
DITTA MURRI - Via A. Diaz, 24/A - 34123 Trieste - Tel. 040.733253

Varese
ELMEC SPA - Via Sebenico, 12 - 21100 Varese - Tel. 0332.264135

Venezia
COMPUTIME SRL - Piazza Rizzo, 63 - 30027 S. Donà di Piave - Tel. 0421.2548

Vercelli
ANALOG SNC - Via Dionisotti, 18 - 13100 Vercelli - Tel. 0161.61105
CENTRO SERVIZI INFORMATICA TEOREMA SRL - Via Losano, 9 - 13051 Biella - Tel. 015.24915

Verona
PRAGMA SOFTWARE SRL - Via Carmelitani Scalzi, 20 - 37100 Verona - Tel. 045.24629

Vicenza
ALFA DATA SRL - Via Milano, 110 - 36042 Vicenza - Tel. 0445.874199

Viterbo
ITALBYTE SRL - V.le Trento - Pal. Garbini 01100 Viterbo - Tel. 0761.221333

- E per acquisti superiori alle 20 unità puoi anche rivolgerti alle filiali IBM.
- Per ulteriori informazioni sugli indirizzi dei punti di vendita telefona a 02/21752360 oppure 06/54864962.

IBM Italia
Distribuzione Prodotti srl

L'input - output digitale con Apple

di **S. Rossi**

È ormai noto che un computer può trattare solo segnali digitali; anche la lettura di segnali analogici deve essere trasformata in qualche modo in digitale da opportuni convertitori.
L'acquisizione di segnali digitali provenienti dai più svariati trasduttori è un compito particolarmente semplice per un computer e qui ci proponiamo di presentare alcuni programmi adatti allo scopo.
Un segnale digitale può assumere solo due valori di tensioni compresi in fasce ben definite. Il valore di tensione più basso lo definiremo, per semplicità, come livello zero, mentre il valore più alto, come livello uno. Nel caso di logiche alimentate a 5 V il livello zero è un valore di tensione inferiore a 0,8 V e il livello uno un valore superiore a 2 V (i valori di tensione variano a seconda della tecnologia costruttiva del componente, ma questo è irrilevante ai fini di quanto esporremo).
Il segnale digitale tipo può essere ottenuto da un circuito come quello riportato in figura 1 e costituito da un interruttore e da una resistenza Quando l'interruttore è aperto si ha una tensione di 5 V; quando l'interruttore è chiuso si ha una tensione di 0 V.
I segnali digitali possono essere agevolmente rilevati dall'interfaccia VIA per APPLE presentata nel numero di gennaio dello scorso anno.
L'interfaccia VIA è fornita, fra l'altro, di due porte parallele bidirezionali da 8 bit, definibili cioè in ingresso o in uscita e capaci quindi di leggere segnali digitali o di inviare messaggi sotto forma digitale o, ancora, di azionare dispositivi come relè o LED.
L'Apple, come qualsiasi calcolatore con microprocessore da 8 bit, può trattare appunto 8 segnali digitali contemporaneamente. Questi 8 segnali digitali hanno un significato (peso) diverso a seconda della loro posizione. La figura 2 mostra come si ricava l'equivalente decimale di un numero binario da 8 bit.
Il bit 0 pesa 1 (2^0); il bit 1 pesa 2 (2^1); il 2 pesa 4 (2^2) e così via fino all'ottavo bit, il bit 7, che pesa 128 (2^7). Per ricavare il valore decimale di un numero di otto bit occorre sommare i pesi dei bit che sono a 1. Collegando alla porta A una serie di 8 interruttori e di 8 resistenze, come quelle prima descritte, è possibile eseguire alcune prove di acquisizione (figura 3).

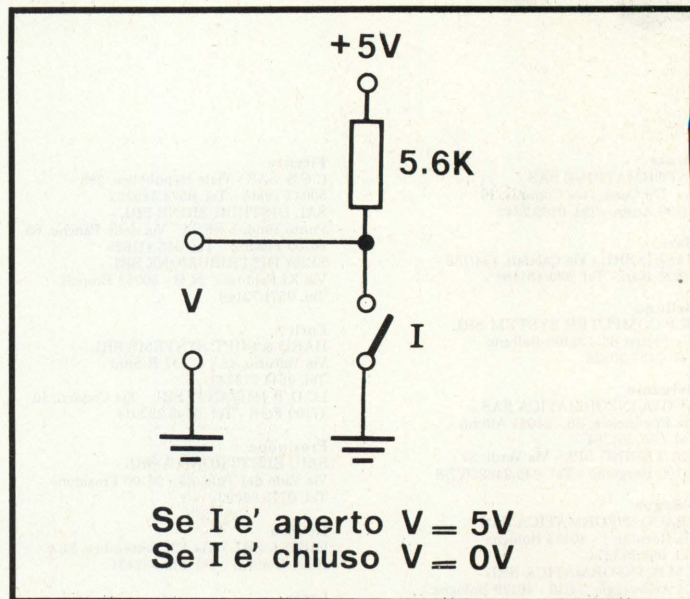



Figura 1 - Il segnale tipo.

bit	7	6	5	4	3	2	1	0	bit
peso	128	64	32	16	8	4	2	1	peso
esempio	0	1	0	0	0	1	1	0	→ 2+4+64 = 70

ovvero:

$$2^7 \star 0 + 2^6 \star 1 + 2^5 \star 0 + 2^4 \star 0 + 2^3 \star 0 + 2^2 \star 1 + 2^1 \star 1 + 2^0 \star 0 = 70$$

Figura 2 - Come si ricava il numero decimale corrispondente a una combinazione di 8 bit.

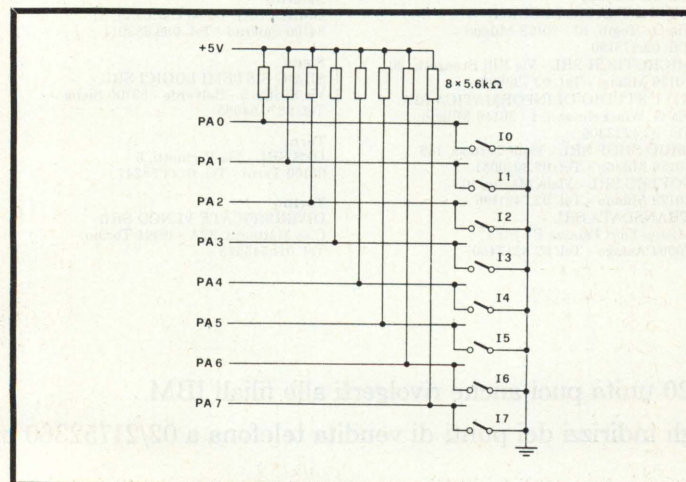



Figura 3 - Un primo esperimento.

Occorre inanzitutto scrivere un programma che "catturi" il dato all'ingresso della porta e lo visualizzi sul monitor Apple.
Un semplice programma adatto allo scopo è il seguente:

```
10  POKE 50675,0
20  PRINT PEEK (50673)
30  FOR I=1 TO 200:NEXT I
40  HOME
50  GOTO 20
```

La riga 10 definisce la porta A in input.
La riga 20 legge il dato e lo visualizza.
La riga 30 è un ritardo per consentire la visualizzazione del dato.

```
5   REM  STAMPA DI UN NUMERO IN FORMA DECIMALE E BINARIA
10  REM  DEFINISCE LA PORTA A IN INPUT
20  POKE 50675,0
30  REM  LEGGE IL DATO
40 D =  PEEK (50673)
45 D1 = D
50  REM  CONVERTE IL DATO BINARIO.IL DATO BINARIO E'IN B$
60 B$ = ""
70  FOR I = 7 TO 0 STEP  - 1
80  IF D / 2 ^ I < 1 THEN B$ = B$ + "0": GOTO 100
90 B$ = B$ + "1"
95 D = D - 2 ^ I
100  NEXT I
110  PRINT "DECIMALE","BINARIO"
120  PRINT D1,B$
130  FOR I = 1 TO 300: NEXT I
135  HOME
140  GOTO 30
```

*Figura 4 - **Rappresentazione della lettura della porta in forma decimale e binaria.***

```
10  REM  CONTAPEZZI
20  REM  DEFINISCE PORTA A IN INPUT
30  POKE 50675,0
40  REM  DEFINISCE VETTORE CONTATORE E VETTORE DI STATO
50  DIM C(7): DIM S(7)
60  REM  LEGGE DATO
70 D =  PEEK (50673)
80  REM  CONVERTE E MEMORIZZA IL DATO INIZIALE
90  FOR I = 7 TO 0 STEP  - 1
100  IF D / 2 ^ I < 1 THEN S(1) = 0: GOTO 130
110 S(I) = 1:C(I) = C(I) + 1
120 D = D - 2 ^ I
130  NEXT I
140  REM  CONTA
145 D =  PEEK (50673)
150  FOR I = 7 TO 0 STEP  - 1
160  IF D / 2 ^ I < 1 THEN S(I) = 0: GOTO 190
170  IF S(I) = 0 THEN S(I) = 1:C(I) = C(I) + 1
180 D = D - 2 ^ I
190  NEXT I
200  HOME : PRINT "MACCHINA","PEZZI"
210  FOR I = 0 TO 7
220  PRINT I,C(I)
230  NEXT I
240  GOTO 145
```

Figura 5 - Programma "Contapezzi".

La riga 40 cancella il visore.
Si faccia partire il programma e si tengano gli interruttori aperti: comparirà il numero 255 (tutti i bit a "1").
Se si chiudono tutti gli interruttori comparirà il numero 0 (tutti i bit a "0"). Se si apre solo l'interruttore corrispondente al bit 0 comparirà il numero 1 (in forma binaria 00000001).
Se si apre anche l'interruttore corrispondente al bit 1 comparirà il numero 3 (in forma binaria 00000011) e così via per tutte le 256 combinazioni possibili.
Dalla prova eseguita è possibile notare che l'istruzione BASIC PEEK richiama il contenuto di una locazione di memoria espresso sotto forma decimale.
Spesso è necessario distinguere quale bit è a zero e quale a uno; questa operazione è semplicissima se si lavora in linguaggio macchina e un pò più complicata e molto più lenta se si programma in BASIC.
Un programma che visualizza il contenuto, sia in forma binaria che in quella decimale, è riportato in figura 4.
Facciamo un ulteriore passo.
Supponiamo di voler realizzare otto contapezzi per altrettante macchine utensili che non producano più di un pezzo al secondo ciascuna e che il segnale di "pezzo finito" abbia una durata di almeno mezzo secondo e sia indicato da un interruttore aperto.
Un programma che risolve il problema è riportato in figura 5.
Il vettore C contiene il numero di pezzi per ogni macchina; il vettore S contiene l'ultimo stato letto.
Non è necessario usare degli interruttori per la variazione del

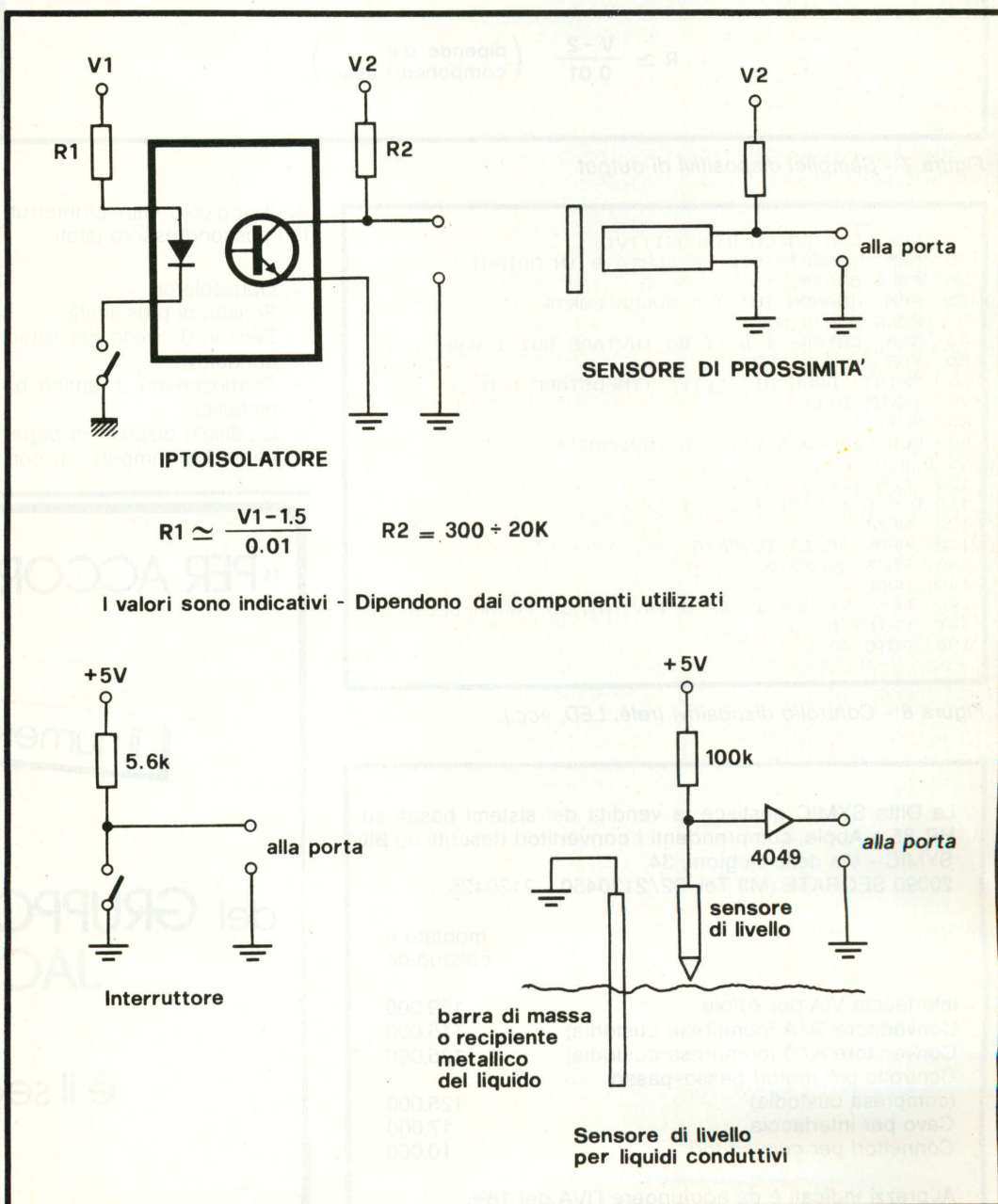



Figura 6 - Tipici circuiti generatori di segnali digitali.

L'input - output digitale con Apple

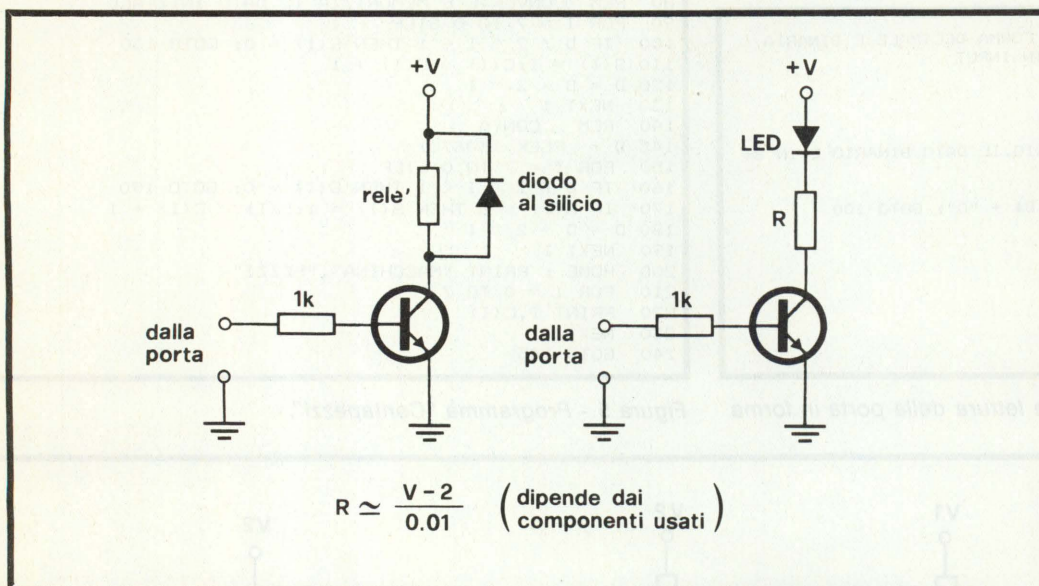



Figura 7 - Semplici dispositivi di output.

```
10  REM  CONTROLLO DISPOSITIVI
20  REM    DEFINISCE LA PORTA B IN OUTPUT
30  POKE 50674,255
35  REM    MANDA TUTTI 0 SULLA PORTA
37  POKE 50672,0
40  REM  CHIEDE I DATI DA INVIARE SUI I VARI BIT
50  FOR I = 0 TO 7
60  PRINT "DARE BIT ";I;" (IMPOSTARE 0 O 1)"
70  INPUT B(I)
80  NEXT I
90  REM  CALCOLA DATO DA INVIARE
95 D = 0
100  FOR I = 0 TO 7
110 D = D + B(I) * 2 ^ I
120  NEXT I
130  REM  INVIA IL DATO
140  POKE 50672,D
145  HOME
146  REM  STAMPA IL DATO INVIATO IN FORMA DECIMALE
147  PRINT D
150  GOTO 40
```

Figura 8 - Controllo dispositivi (relè, LED, ecc.).

livello logico 0. Oltre gli interruttori possono essere usati:

- Optoisolatori
- Sensori di prossimità
- Sensori di livello per liquidi conduttivi
- Termosensori a lamina bimetallica
- Qualsiasi dispositivo capace di interrompere un corrente.

La figura 6 riporta alcuni tipici circuiti.
L'interfaccia VIA per APPLE ha anche un'altra porta, la porta B, che può essere usata per chiudere relè, accendere led o pilotare altri dispositivi.
La figura 7 mostra come collegare un relè o un Led alla porta. Il transistor deve essere del tipo Darlington, vale a dire ad elevato guadagno, perchè le correnti erogabili dalla porta sono basse, dell'ordine del milliampere.
Un semplice programma per pilotare la porta B è riportato in figura 4.
Se alla richiesta del bit si imposta 1 si ecciterà il relè o si accenderà il LED collegato al bit relativo; se si imposta 0 il dispositivo rimarrà inattivo.
La semplice gestione di input-output descritta consentirà, in ambiente tecnico, di controllare alcune funzioni di macchine utensili e, in ambiente amatoriale, di realizzare contagiri per piste elettriche, comandi per trenini, antifurti, annaffiatoi automatici e comunque di comprendere un po' di più le possibilità del vostro APPLE. ■

Semaforo cibernetico

PRESENTAZIONE

Questo articolo rappresenta la "sofferta" riduzione del lavoro svolto nell'anno scolastico 1981-1982 da tre alunni di V classe dell'indirizzo automatico elettronico nell'I.T.C. "A. Ruiz" di Roma. Speriamo di riuscire a fornire tutte le informazioni necessarie per la riproduzione didattica di tale lavoro, visto che la sintesi per la pubblicazione, senza dubbio utile come esperienza, ha rappresentato una ulteriore difficoltà. Ci sembra comunque importante non trascurare la concretezza del tema svolto e le soluzioni prospettate. Tanto per smentire il luogo comune secondo il quale a scuola ciò è impossibile, senza cadere in temi banali.
I coordinatori: Prof. Giovambattista Garsia e Prof. Lorenzo Severini.

di **A. Capizzi, R. Fortura, P. Giangregorio**

Il nostro progetto di controllo di semaforo si inserisce tra le varie pubblicazioni su tale argomento in modo sostanzialmente nuovo.
Si tratta infatti di un semaforo "intelligente" in cui l'utilizzazione del computer permette l'alleggerimento dell'hardware, ma soprattutto offre la possibilità di adattare il funzionamento del semaforo alle variazioni del mondo esterno.
Il sistema, basato sul nanocomputer NBZ80S, prevede oltre alle normali routine dei tempi e degli stati, anche due condizioni di emergenza che si creano nella realtà e che normalmente vengono risolte dalla presenza di un vigile urbano:
a) il flusso del traffico in una direzione è molto maggiore di quello nell'altra;
b) il sopraggiungere di una vettura con sirena in funzione (pompieri, polizia, ambulanza).
Nel caso a), il sistema modifica i tempi di accensione degli stati Rosso-Verde nei due sensi.
Nel caso b), esso forza lo stato di verde nella direzione che lo necessita.
Nello studio del progetto abbiamo deciso di ricorrere, ogni qual volta fosse possibile, a soluzioni software che limitando l'hardware, riducono il costo

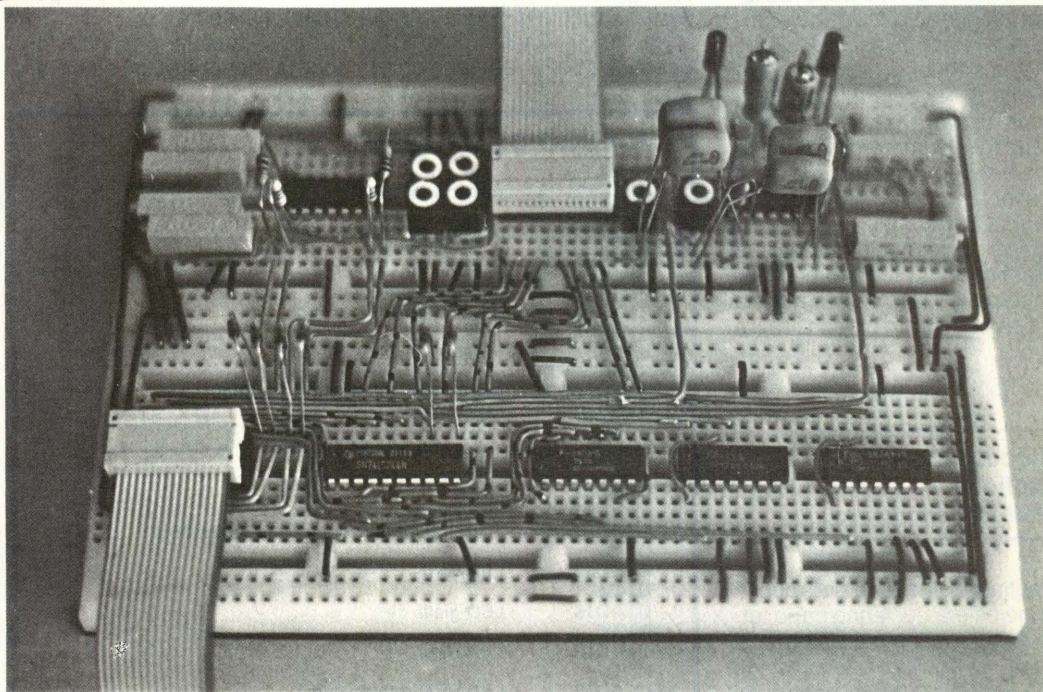



Foto 1 - Breadboard.

Foto 3 - Basetta montata.

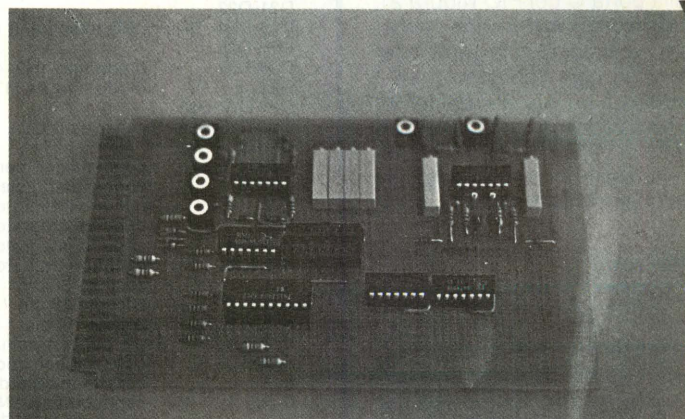
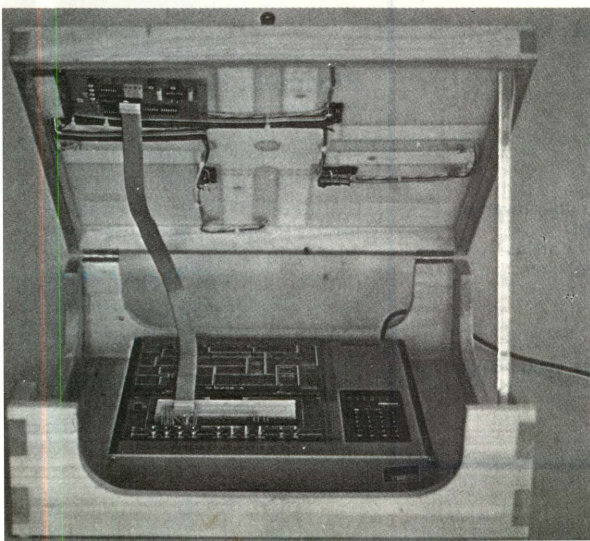
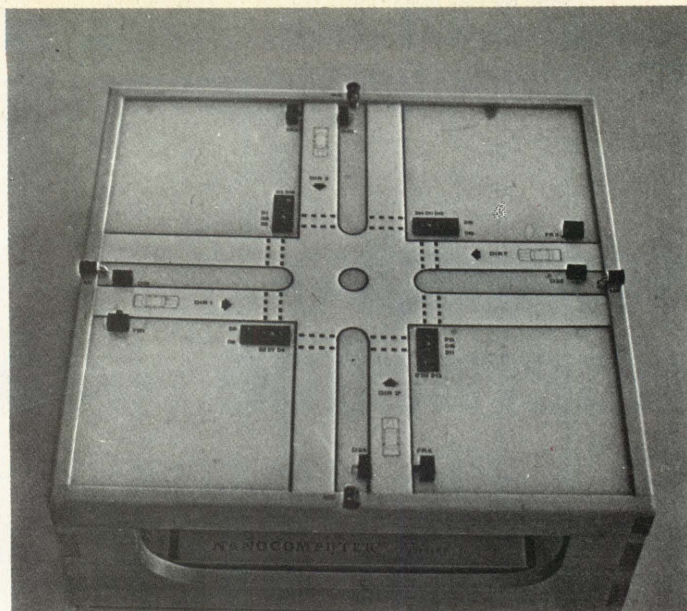



Foto 2 - Disposizione dei componenti all'interno del plastico.

Semaforo cibernetico

della realizzazione. Per rendere il semaforo intelligente abbiamo utilizzato dei rivelatori ottici (passaggio vetture) e dei rivelatori acustici (vetture con sirena) disposti nei quattro versi come si vede dalla foto del plastico. I segnali dei rivelatori opportunamente elaborati, hanno la funzione di generare comandi di interrupt.
In seguito agli stati del semaforo verranno indicati con *ROS*, *GIA*, *VER*, seguiti da un 1 o 2 che definiscono la direzione e un verso di percorrenza, mentre 1' e 2' definiscono i rispettivi versi opposti.
Vediamo lo schema di flusso del sistema e la relativa descrizione dei blocchi (figura 1). Parliamo prima di tutto della realizzazione del semaforo "convenzionale" per poi inserire le parti che lo rendono "cibernetico".
Diamo per scontata la successione delle diverse configurazioni di un semaforo, salvo dire che il verde nelle due direzioni presenta una temporizzazione pari a 30 secondi.
Passiamo immediatamente ad esaminare i programmi e gli schemi. Il sistema tradizionale si compone di tre blocchi: A, B e H.

Blocco A: routine gestione stati del semaforo

Routine *START*, è la routine principale a cui fanno capo tutte le subroutine del sistema. Si basa sull'utilizzo dello stack (*ICON*), in cui sono contenute le configurazioni da presentare sulla porta C del PIO (figura 2), e dello stack in cui sono contenuti i valori dei tempi corrispondenti ad ogni configurazione. I contenuti dello stack dei tempi inizializzano un contatore che viene di volta in volta diminuito, e prima di ogni decremento avviene una chiamata alla routine *DELAY*.
Ripetendo questo procedimento fino ad azzerare il contatore, si ottiene il mantenimento sulla porta C delle particolari configurazioni per il tempo desiderato.
Routine *GIALL*, viene abilitata quando l'orologio (routine *DAY*) è azzerato, ossia quando è terminato il funzionamento diurno, della durata di 16 ore. Durante la sua esecuzione gli interrupt sono disabilitati, il clock alla frequenza di 2 Hz è interrotto e sulla porta C si presentano alternativamente le configurazioni Giallo1 - Giallo2 Spento1-Spento2 (giallo lampeggiante) con frequenza di 1 Hz. La durata del ciclo notturno è di 8 ore.
Routine di ritardo *DELAY*, genera un ritardo di 0.409 secondi e viene chiamata dalla maggior parte dei blocchi che compongono il sistema.
Questa routine provvede anche a generare un clock alla frequenza di 2 Hz, che viene utilizzato in seguito nel blocco F. Non è stato necessario tenere conto dei tempi macchina in quanto irrilevanti ai fini del ritardo.
Routine orologio *DAY*: giorno-notte, viene chiamata dalla routine DELAY, vale a dire ogni 1/2 secondo e determina l'inizio e la fine del funzionamento diurno e notturno. Il criterio adottato consiste nell'uso di tre locazioni di memoria come contatori nidificati.
Questi vengono caricati con opportuni valori e quindi decrementati fino all'azzeramento. La verifica di fine ciclo non avviene in questa routine ma nel programma chiamante. Ciò è possibile in quanto l'istruzione *RET* non modifica lo stato dei Flag.

ELENCO COMPONENTI

R3-R4-R9	
R10-R35	: resistori da 100 Ω
R1-R2-R5	
R6-R7-R8	: resistori da 200 Ω
R13-R14-R15	
R16-R29-R30	: resistori da 1 K Ω
R25-R26	: resistori da 300 K Ω
R27-R28	: resistori da 1 K Ω trim.
R17-R18	
R19-R20	: resistori da 25 K Ω trim.
R21-R22-R23	
R24-R33-R34	: resistori da 100 K Ω trim. multigiri
R31-R32	: resistori da 1 M Ω

Tutte le resistenze sono da 1/4 W

FR1-FR2	
FR3-FR4	: fotoresistenze
M1-M1'-M2-M2'	: microfoni unidirezionali a condensatore
D1-D2	
D11-D12	: LED verdi
D7-D8	
D17-D18	: LED gialli
D4-D5	
D14-D15	: LED rossi
D3-D6	
D13-D16	: LED rossi rettangolari
D9-D10	
D19-D20	: LED verdi rettangolari
D21-D22	
D23-D24	: LED trasparenti rettangolari a luce gialla
D25-D26	
D28-D29	: diodo tipo 1N914
D27	: diodo tipo AA117
Dz1-Dz2-Dz3	
Dz4-Dz5-Dz6	: zener da 5,1 V
C1-C2	
C3-C4	: condensatori ceramici 22 KpF
C5-C6	: condensatori elettrolitici 1 μF
IC1	: circuito integrato 74LS240 Octal buffer inverter
IC2	: circuito integrato 7400 Quad NAND
IC3-IC4	: circuito integrato 7490 BCD caunt.- divid.
IC5-IC6	: circuito integrato LM 349 Quad op. amp.
n. 1	: zoccolo 20 pin
n. 1	: zoccolo 18 pin
n. 5	: zoccoli 14 pin
n. 2	: pin per test point
n. 1	: cavo flat con zoccoli dip a 18 linee, lungo 80 cm.
n. 1	: basetta eurocard prestampata
n. 1	: nanocomputer NBZ80-S
n. 1	: registratore RCZ80
n. 1	: cassetta audio C30

Blocco H, interfacciamento

L'interfacciamento del computer con i rivelatori avviene tramite la porta PIO (input-output parallelo), a disposizione sul NBZ80-S.
Il PIO contiene due porte, C e D, che utilizziamo in due diversi modi. La porta C (figura 3a) è programmata in modo 0 (uscita di un byte) e fornisce al buffer le configurazioni (*VER*, *GIA*, *ROS*) da visualizzare.
La porta D (figura 3b) è programmata in modo 3 (controllo dei singoli bit) e verrà utilizzata per comunicare al nanocomputer la presenza delle condizioni di emergenza (A e B).

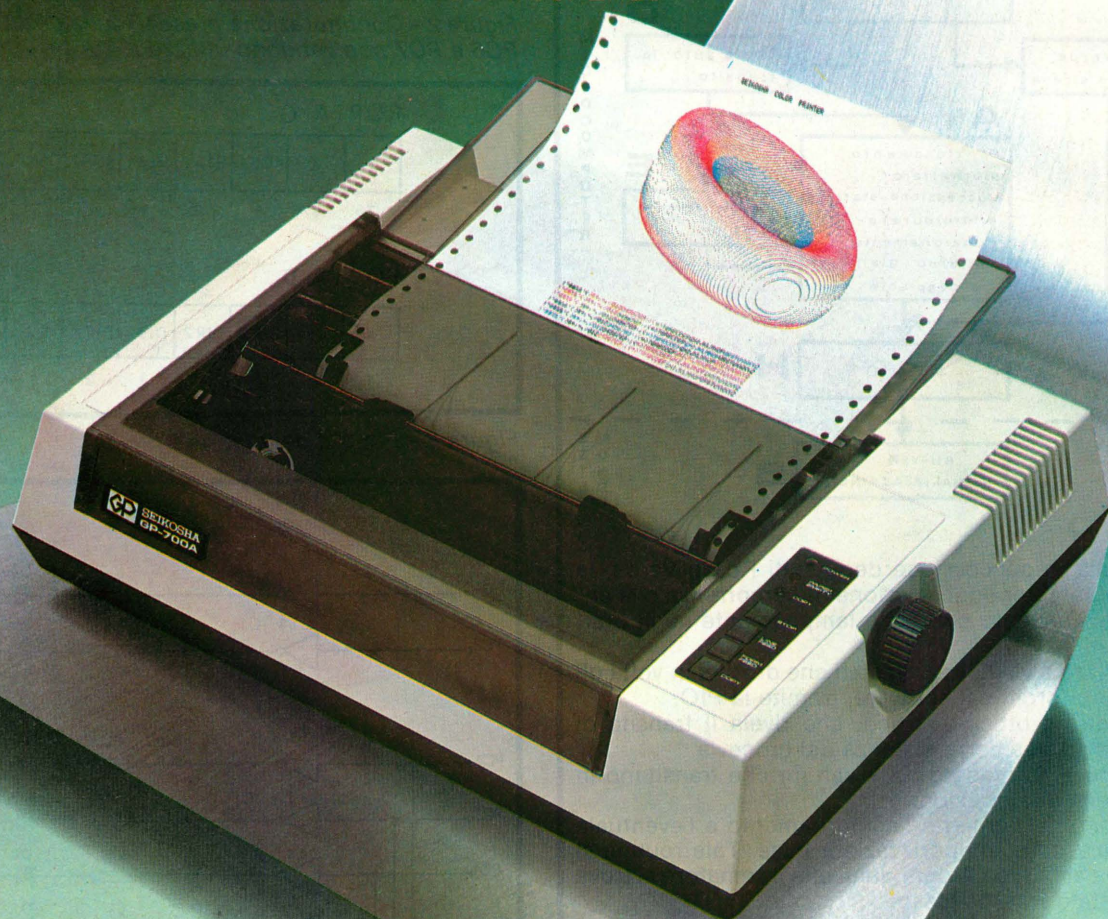
Blocco B, buffer

Poiché il fan-out delle PIO è inferiore a quello necessario, abbiamo utilizzato sei buffer invertenti LS per pilotare i led che simulano le luci del semaforo (figura 4). Le quattro porte NAND, permettono il corretto funzionamento dei led per i pedoni *VERDE P1 VERDE P2* (figura 5). *ROSSO P1* e *ROSSO P2* sono ricavati direttamente dagli stati *ROSSO 2* e *ROSSO 1*.

QUANTI COLORI HA LA TUA STAMPANTE ?

NEL 1983 LA SEIKOSHA PER PRIMA AL MONDO
E' IN GRADO DI PRESENTARE LA NUOVA STAMPANTE
GRAFICA A SETTE COLORI.

RIUNITE IN UN APPARECCHIO PRATICO E COMPATTO
LE CARATTERISTICHE DELLA STAMPANTE E DEL PLOTTER,
LA SEIKOSHA INVENTA UN NUOVO TIPO DI PERIFERICA
CHE BEN PRESTO SARA' INSOSTITUIBILE.

REBIT COMPUTER E' ORGOGLIOSA DI LANCIARE
QUESTA NOVITA' ASSOLUTA SUL MERCATO ITALIANO
AD UN PREZZO MOLTO, MOLTO COMPETITIVO:
MENO DI UN MILIONE.
MENO DI UNA COMUNE STAMPANTE IN BIANCONERO.

GP-700A

Graphic Color Printer

SEIKOSHA

MATERIALE PER LA REALIZZAZIONE DEL PLASTICO

n. 1	: lastra plexiglass 50 x 50 cm
n. 1	: box in legno 52 x 52 x 20 cm
n. 1	: connettore femmina a 21 poli
n. 4	: supporti per microfoni
n. 4	: supporti per led trasparenti
n. 4	: supporti per fotoresistenze
n. 4	: cannoncini direzionali per fotoresistenze
n. 4	: supporti per semaforo
—	: trasferibili vari e carta adesiva
—	: cm 200 di cavetto schermato
—	: cm 200 di piattina a 15 linee colorate.

Emergenza A

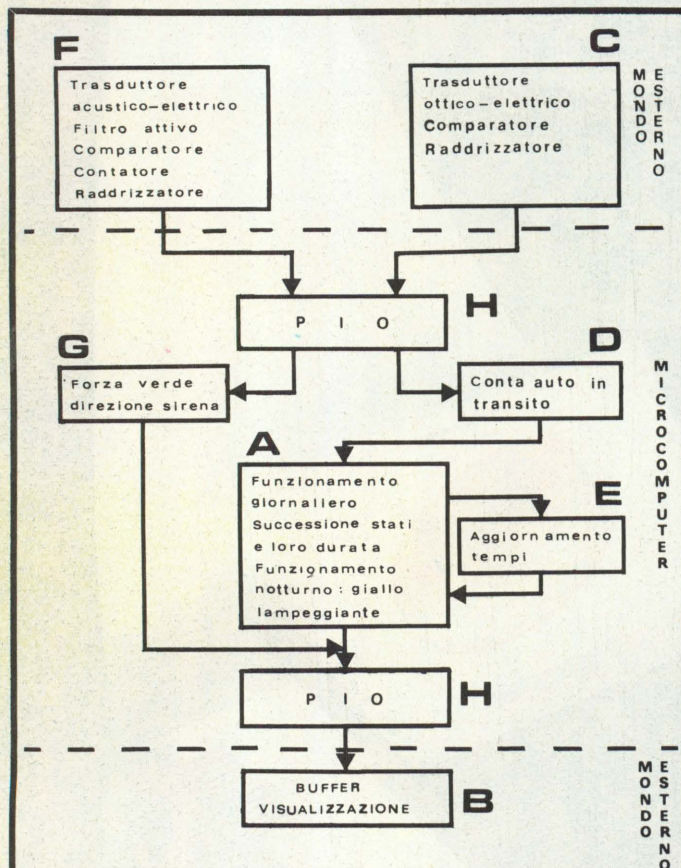
Parliamo ora della prima integrazione che qualifica il nostro sistema. Consiste nella valutazione del flusso stradale esistente nelle due direzioni. Abbiamo ritenuto che sia possibile segnalare il passaggio delle ruote delle vetture ponendo una coppia formata da una sorgente luminosa e da un rivelatore. Le quattro coppie sono sistemate ai bordi opposti di ciascuna corsia ad una certa distanza dall'incrocio. Elaborando questa rilevazione è possibile variare le temporizzazioni degli stati Rosso-Verde in modo da

A) Gestisce la successione degli stati Rosso-Verde, la loro durata per ogni direzione di percorrenza e la entrata in funzione del giallo lampeggiante negli orari notturni.
B) Bufferizza e visualizza gli stati che di volta in volta gli vengono forniti dal software, tramite la PIO.
C) Trasduttore ottico-elettrico, che rivela il transito di vetture ad una certa distanza dal crocevia.
D) Routine di conteggio delle vetture che transitano in una delle due direzioni.
E) Routine che permette l'aggiornamento e l'eventuale modifica delle temporizzazioni base. Tale routine si collega al blocco A ogni qualvolta il semaforo si trova nello stato Rosso-Rosso.
F) Trasduttore acustico-elettrico, che rivela il sopraggiungere di una vettura con sirena in funzione.
G) Routine che forza il semaforo nello stato di Verde nella *direzione in* cui è segnalata la presenza di una sirena in funzione.
H) PIO: permette l'interfacciamento del microprocessore con il mondo esterno.

Figura 1 - Schema di flusso e descrizione dei blocchi.

STATI	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0	
	X	X	ROS1	GIA1	VER1	ROS2	GIA2	VER2	HEX
ROSSO1-VERDE2	0	0	1	0	0	0	0	1	21
ROSSO1-VERDE2/GIALLO2	0	0	1	0	0	0	1	1	23
ROSSO1-ROSSO2	0	0	1	0	0	1	0	0	24
VERDE1-ROSSO2	0	0	0	0	1	1	0	0	OC
VERDE1/GIALLO1-ROSSO2	0	0	0	1	1	1	0	0	1C
ROSSO1-ROSSO2	0	0	1	0	0	1	0	0	24
GIALLO1-GIALLO2	0	0	0	1	0	0	1	0	12
SPENTO1-SPENTO2	0	0	0	0	0	0	0	0	OO

Figura 2 - Configurazione presenti sulla porta C del PIO. I bit PC6 e PC7 non vengono utilizzati.

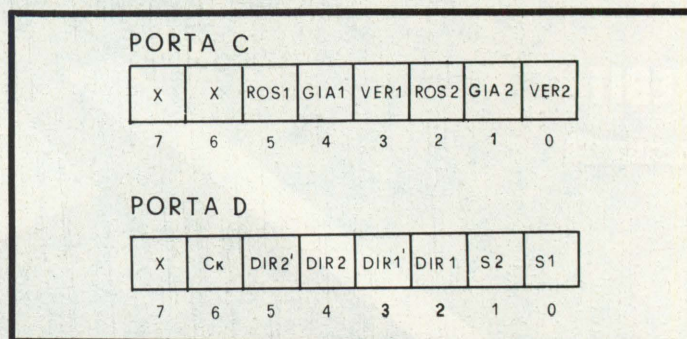



Figura 3 - Pin-out delle porte.

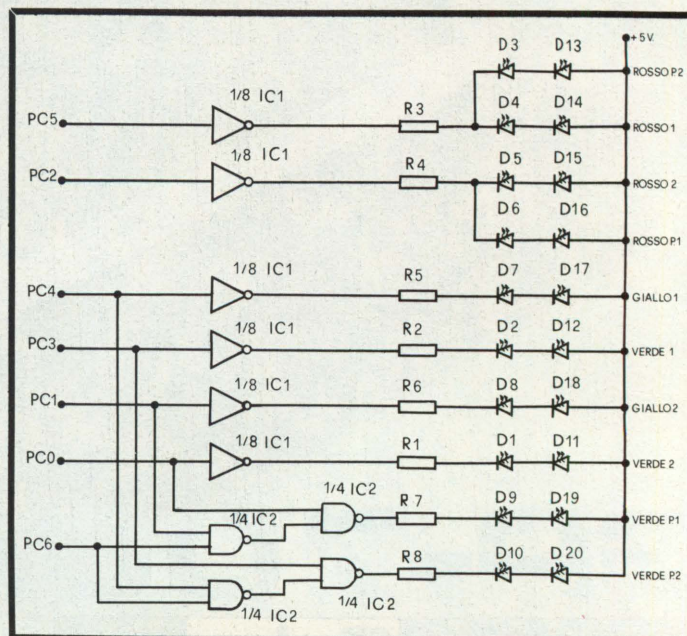



Figura 4 - Buffer e visualizzatori.

personal video monitor 10"

A COLORI

- **Ingresso PAL · AUDIO**
- **Alta definizione**
- **Amplificatore audio incorporato**
- **Facilmente collegabile a tutti gli home, personal computer e videogame**

e vedi meglio il video sempre tuo!

REBIT
COMPUTER
A DIVISION OF G.B.C.

TABELLA VERITA:

VER1	GIA1	CK	VERP2
0	0	0	1
0	0	1	1
0	1	0	x
0	1	1	x
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$$VERP2 = \overline{VER1} + GIA1 \cdot CK = \overline{VER1 \cdot \overline{GIA1 \cdot CK}}$$

$$VERP1 = \overline{VER2} + GIA2 \cdot CK = \overline{VER2 \cdot \overline{GIA2 \cdot CK}}$$

Figura 5 - Funzionamento dei led pedoni.

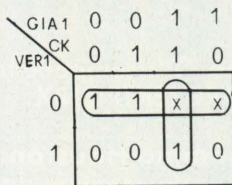
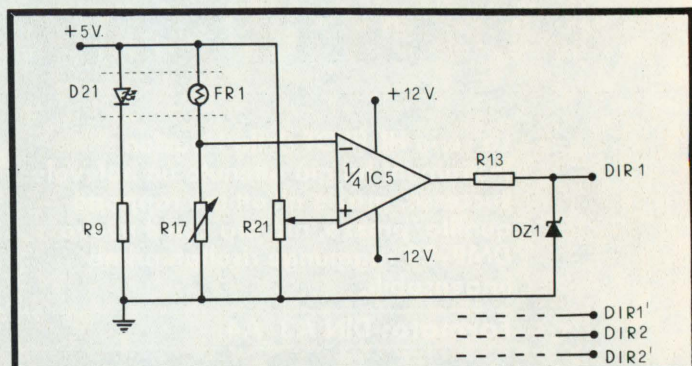



Figura 6 - Rivelatore passaggio vetture.

far defluire opportunamente il traffico.
Questo è l'obiettivo dei blocchi C, D ed E che andiamo ad esaminare.

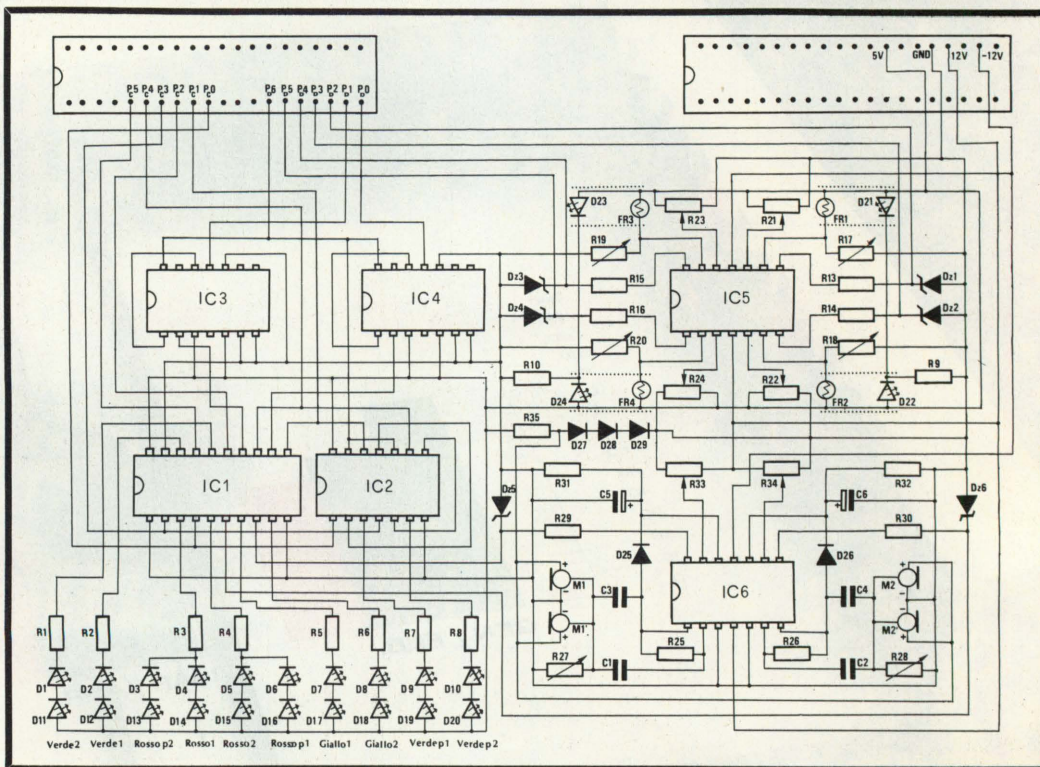
Blocco C: rivelatore passaggio vetture

Nel blocco C (figura 6), vi sono 4 stadi uguali, uno per ogni senso di marcia. Descriviamo il verso DIR1. La fotoresistenza FR1 è collegata all'ingresso invertente di uno dei quattro operazionali contenuti nell'integrato LM 349. Ogni Op. Amp. montato come comparatore, confronta il segnale in esame (ramo fotoresistenza) con quello di riferimento preso dal centrale del trimmer multigiri R21. Se la fotoresistenza è colpita dal fascio di luce emessa da D21, cioè non sta passando alcuna vettura, si comporta da elemento conduttore. La tensione ai capi di R17 (ingresso invertente), è maggiore di quella di riferimento (ing. non invertente) e il comparatore avrà in uscita una tensione negativa.
Se la fotoresistenza non è colpita dal fascio di luce, cioè sta passando una vettura, offrirà una elevata resistenza. Dunque la tensione ai capi di R17 sarà minore del livello di riferimento e l'operazionale avrà in uscita una tensione positiva.
Affinché i livelli assunti in uscita dal comparatore siano compatibili con gli stati logici digitali, abbiamo posto in uscita il diodo zener Dz1 (5,1V), con rispettiva resistenza di caduta R13.
Le quattro uscite Dir1, Dir1', Dir2, Dir2' sono collegate alla porta D della PIO.

Blocco D: routine di conteggio

La routine *RIVEL* viene abilitata ogni qualvolta la porta D della PIO genera una interruzione.
Procede testando i 6 bit non mascherati della porta D, dal meno al più significativo.
I rivelatori di transito vetture, sono collegati ai bit PD2, PD3, PD4, PD5 e vengono utilizzati due registri della CPU (B' e C') come contatori vetture per i 2 versi di una direzione. Inizialmente controlliamo lo stato at-

Figura 7 - Rivelatore acustico.

Figura 8 - Schema elettrico complessivo.

Sistema Philips PMDS
Un nuovo sistema che cresce con i vostri programmi di sviluppo.

- Unità a disco rigido
- Potente sistema operativo basato su UNIX
- Possibilità multiuser
- Supporto completo per 8 e 16 Bit
- Estendibilità e upgrading

Il nuovo sistema di sviluppo universale Philips, PM 4422, si affianca sul mercato al ben noto PM 4421 (sistema universale a doppio floppy) con aumentata potenza e flessibilità pur mantenendone compatibilità hardware e software.

• Un investimento sicuro
Il PM 4422 partendo da una configurazione monoutente, può essere espanso, in configurazione multiutente, fino ad un max di 7 stazioni di lavoro. Così con il crescere dei vostri programmi di sviluppo, il PM 4422 può crescere con essi.

• Memoria di massa su Hard-Disk
La memoria di massa è costituita da un disco rigido Winchester da 5.1/4" inserito direttamente nel sistema, con una capacità di 5 o di 21 Mb. Questa capacità può essere aumentata fino a 147 Mb con l'aggiunta di ulteriori unità a disco.
La memoria interna di sistema può essere espansa da una configurazione standard di 256 Kb ad un max di 1 Mb.

• Sistema operativo basato sullo UNIX*
Il sistema operativo implementato sul PM 4422 è basato sullo UNIX* della Bell di cui son ben note le possibilità multiuser e multitask.

• Supporto 8 e 16 Bit
Il PM 4422 è un sistema di sviluppo universale che supporta microprocessori a 8 e 16 Bit con emulazione in tempo reale.

• Upgrading
I PM 4421 a doppio floppy già esistenti sul mercato possono essere modificati facilmente mediante opportuno kit di conversione in modo da includervi tutti i vantaggi del PM 4422.

* UNIX è un marchio della Bell Laboratories

Philips S.p.A. - Divisione S&I
Strumentazione & Progetti Industriali
Viale Elvezia, 2 - 20052 Monza
Tel. (039) 3635.240/8/9
Telex 333343

Filiali: **Bologna** (051) 493.046
Cagliari (070) 666.740
Padova (049) 632.766
Palermo (091) 527.477
Roma (06) 33.02.344/5/6/7
Torino (011) 21.64.121

PHILIPS

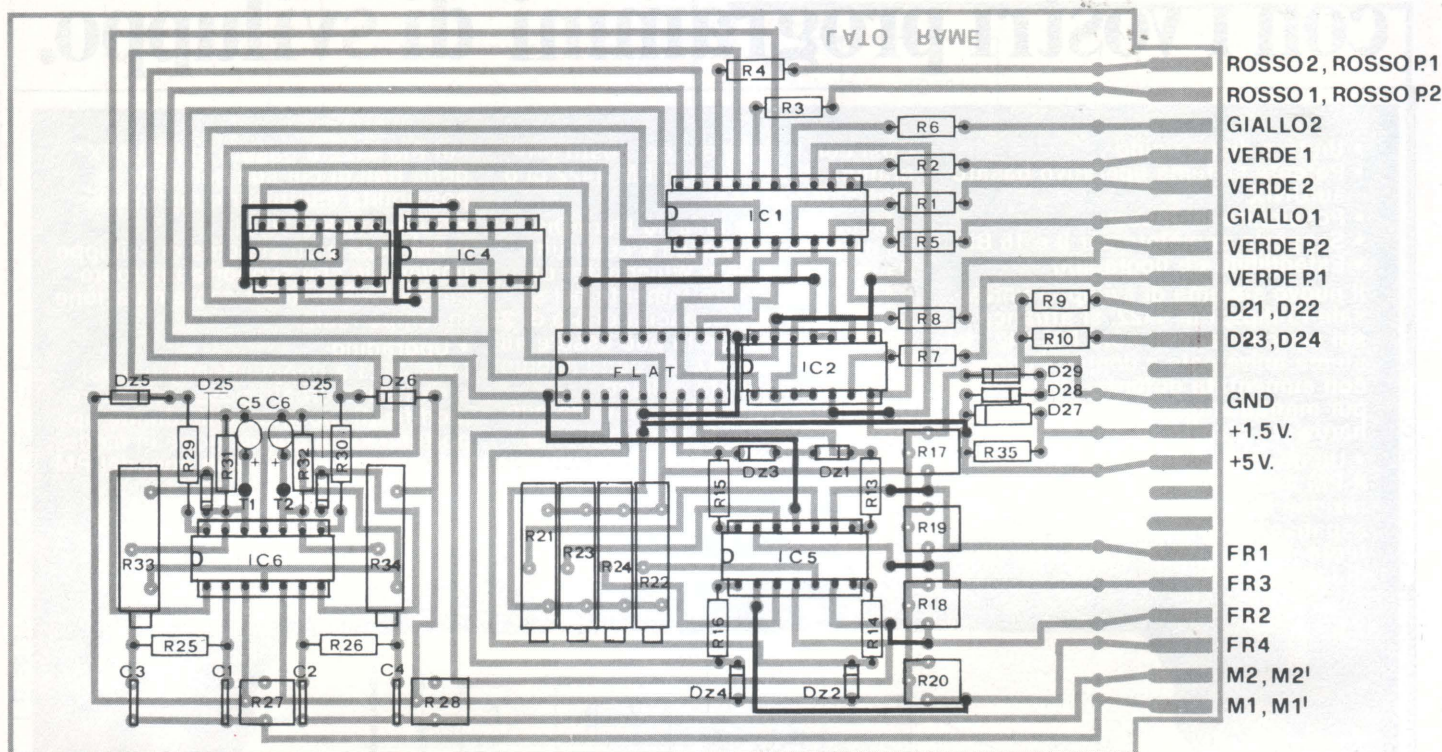



Figura 9 - Disposizione dei componenti.

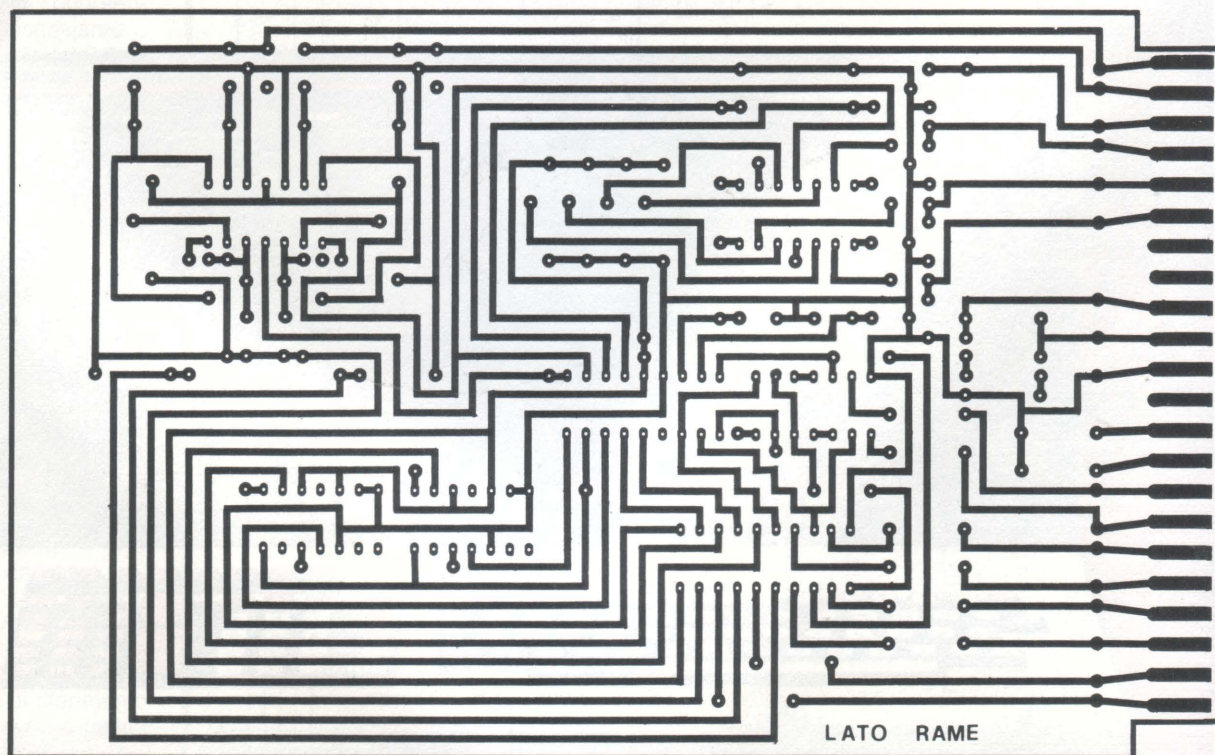
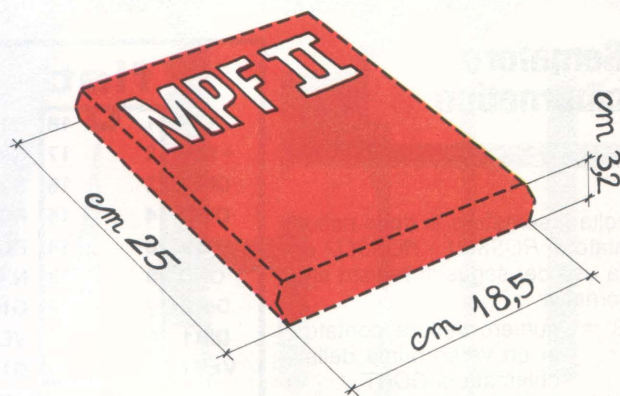



Figura 10 - Cablaggio: lato rame.

tuale del semaforo, dato che nella configurazione *ROSSO1-ROSSO2*, non si debbono contare le vetture in transito tramite il test del bit PC0 della configurazione attuale del semaforo (contenuta nel registro A della CPU), viene deciso se testare i bit PD2, PD3 per la direzione 1, oppure i bit PD4, PD5 per la direzione 2. Quando si trova un bit alto si incrementa il contatore vetture del verso corrispondente. Ad ogni stato *ROSSO1 - ROSSO2*, i contatori vetture vengono azzerati dalla routine *CONT*.

Blocco E: routine aggiornamento tempi

La routine *CONT* di gestione dei tempi è abilitata ogni qual-

1480 cm^3

di MICRO-PROFESSOR MPF II

contengono CPU R6502 - 64 K Bytes di RAM
16 K Bytes di ROM con Interprete Basic Apple Soft

Il MICROPROFESSOR II (MPFII) è un computer unico nel suo genere perché unisce a grandi capacità di memorie residenti **(64 K Bytes di RAM e 16 K Bytes di ROM)** una configurazione di sistema ridottissima.
È veramente portatile.
Le sue minime dimensioni (cm 25 × 18,5 × 3,2) non gli impediscono però di essere un "personal computer" perché oltre ad essere dotato di eccezionali capacità di memoria residenti può essere completato ed allacciato con diverse periferiche.
MPFII diventa così un computer gestionale come altri computer più famosi ed "ingombranti" di lui.
Il modulatore RF e la scheda PALCOLOR residenti vi permetteranno di collegarlo al vostro televisore.
Ecco perché MPFII non è solo "lavoro", ma anche relax.
Insomma un computer idoneo per tutti, dai 7 ai 70 anni di età.
L'ampia disponibilità di software in cassetta, dischi e cartuccia (cartridge) costituisce l'elemento preponderante che lo rende indispensabile come: **SUPPORTO GESTIONALE** (amministrazione, magazzino, acquisti, commerciale, ecc.) per negozi, uffici, aziende. **SUPPORTO SCIENTIFICO PRATICO** per tecnici, professionisti, ricercatori, hobbysti. **SUPPORTO DIDATTICO** per studenti. **SUPPORTO RICREATIVO** (giochi, quiz, ecc.) per tutti.

1) Computer
2) Interfaccia per disk drive
3) Disk drive (slim line)
4) Tastiera esterna

DIGITEK COMPUTER

Ufficio Vendite
Via Marmolada, 9/11 43058 SORBOLO (Parma)
Tel. 0521/69635 Telex 531083

Ag. advert Reggio E.

volta il semaforo si trova nello stato di *ROSSO1 - ROSSO2* e fa uso dei seguenti registri alternativi:

B' = numero vetture contate in un verso prima della chiamata di *CONT*;

C' = numero vetture contate prima della chiamata di *CONT* nel verso opposto a B';

D' = contiene il maggiore dei numeri presenti in B' e C';

H' = numero maggiore vetture contate precedentemente nell'altra direzione;

L' = 20H, scarto minimo per l'aggiornamento dello stack dei tempi.

I registri B' e C' sono gestiti dalla routine di conteggio. Il registro H' è azzerato all'inizio di ogni ciclo diurno e al termine di ogni esecuzione di *CONT* riceve il numero contenuto in D'.

Il compito di questa routine è quello di elaborare i dati provenienti dal blocco hardware (rivelatori passaggio vetture) stabilendo il modo in cui aggiornare i tempi delle varie configurazioni. L'effettivo aggiornamento si ottiene modificando il contenuto del registro IX, il quale punta alla prima locazione di memoria dello stack prescelto, tra i tre possibili (tabella 1).

NORMS: tempi uguali in entrambe le direzioni
MAGGS: tempi maggiori in una direzione
MINS: tempi minori in una direzione.

Essi sono utilizzati separatamente secondo le esigenze del flusso stradale e lo stack preso in esame viene considerato come un vettore. Infatti l'istruzione LD B,(IX+d) viene continuamente modificata considerando "d" come un indice. In questo modo si evita di riscrivere più volte le temporizzazioni aggiornate su una stessa zona buffer di memoria, con un notevole incremento di istruzioni.

Abbiamo già descritto i blocchi che realizzano il semaforo "convenzionale", quelli che gestiscono l'emergenza A (il flusso del traffico in una direzione è molto maggiore di quello nell'altra) e il listato del programma.

Iniziamo ora la descrizione dell'emergenza B (il sopraggiungere di una vettura con sirena in funzione), dello schema elettrico completo e delle nostre proposte di cablaggio.

flat

-12 V.	1	18	S1
+12 V.	2	17	Ck
DIR2'	3	16	S2
DIR1'	4	15	ROS 2
+5 V.	5	14	ROS 1
GND	6	13	N.C.
DIR2	7	12	GIA 2
DIR1	8	11	VER 1
VER2	9	10	GIA 1

Figura 11 - Piedinatura degli zoccoli del cavo flat.

Emergenza B

È il caso in cui sopravviene una vettura dotata di sirena in funzione. Ciò ha creato l'esigenza di dover selezionare segnali acustici.

A tal fine ci siamo serviti di un filtro attivo passa-banda molto selettivo, con banda centrata sulla frequenza della nostra sirena (circa 4 KHz).

Inoltre per evitare comandi indesiderati provocati da suoni occasionali, abbiamo creato una selezione temporale in cui solo la permanenza per almeno 4 secondi del suono rivelato fornisce al sistema il comando opportuno. Vediamo dunque nel dettaglio i blocchi F e G che realizzano quanto detto.

Blocco F: rilevatore vettura con sirena

Nel blocco F (figura 7), vi sono due stadi uguali, uno per ogni direzione. Descriviamo la direzione 1. Vi sono due capsule microfoniche unidirezionali a condensatore (M1, M1') collegate in parallelo, una per ogni verso di percorrenza.

Le capsule vengono alimentate ad 1,5 V sfruttando la caduta sui diodi D27, D28, D29 (1 germanio + 2 silicio), che rispondono con una buona stabilizzazione alla richiesta di corrente estremamente variabile delle capsule. I segnali dei microfoni sono dunque inviati ai filtri attivi passa-banda a circuito risonante RC del 2' ordine.

```
LOC   OP CODE      LABEL   MNEM  OPER       COMMENT

0100  ED 5E        START:  IM    2          ;Modo 2 di interruzione
0102  3E 05                LD    A,05H      ;Carica in I la parte
0104  ED 47                LD    I,A        ;alta dell'indirizzo
0106  31 40 05             LD    SP,0540    ;Carica ind. stack
0109  CD 40 04             CALL  PCO        ;Routine abilitazione
                                            ;porta C
010C  CD 50 04             CALL  PDI        ;Routine  abilitazione
                                            ;porta D
010F  21 40 01             LD    HL,IND     ;Indirizzo indice vett.
0112  3E 40                LD    A,40H      ;clock (bit 6 alto)
0114  32 25 05             LD    (LAC),A    ;
0117  3E FF                LD    A,FFH      ; inizializzazione
0119  32 26 05             LD    (PIM),A    ;
011C  3E E1                LD    A,E1H      ; orologio
011E  32 27 05             LD    (PAM),A    ;
0121  3E 02        TAN:    LD    A,02H      ; giornaliero
0123  32 28 05             LD    (PUM),A    ;
0126  D9                   EX    X          ;Scambia coppie registri
0127  26 00                LD    H,00H      ;Carica H =00
0129  44                   LD    B,H        ;Carica H in B
012A  4C                   LD    C,H        ;Carica H in C
012B  2E 20                LD    L,20H      ;Carica L =20
012D  D9                   EX    X          ;Scambia coppie registri
012E  16 00                LD    D,00H      ;Carica D =00
0130  DD 21 00 05          LD    IX,NORMS   ;Carica in IX indirizzo
                                            ;locazione temporizzazioni
0134  36 FF        TUN:    LD    (HL),FFH   ;Inizializza l'indice vett.
0136  FD 21 14 05          LD    IY,ICON    ;Carica in IY indirizzo
                                            ;locazione configurazioni
013A  F3           INIT:   DI               ;Disabilita l'interrupt
013B  34                   INC   (HL)       ;Incrementa indice vett.
013C  FD 23                INC   IY         ;Incrementa ind. config.
013E  DD 46 00             LD    B,(IX+d)   ;Carica il puntato di
                                            ;IX in B
0141  FD 7E 00             LD    A,(IY+00)  ;Carica il puntato di
                                            ;IY in A
0144  FB                   EI               ;Abilita l'interrupt
0145  D3 08                OUT   (08H),A    ;Pone A in uscita sulla
                                            ;porta C
0147  CD 00 04     TIN:    CALL  DELAY      ;Salta alla routine di
                                            ;ritardo
014A  CA 60 01             JP    Z,GIALL    ;Se (PUM)=0 salta a Giall
014D  05                   DEC   B          ;Decrementa B
014E  C2 47 01             JP    NZ,TIN     ;Se B>0 salta a TIN
0151  3E 24                LD    A,24H      ;Carica A =24 configura-
                                            ;zione R1 - R2
0153  FD 96 00             SUB   (IY+00)    ;Sottrae ad A la confi-
                                            ;gurazione presente in
                                            ;uscita
0156  CA 00 02             JP    Z,CONT     ;Se =0 salta a CONT
0159  C3 3A 01             JP    INIT       ;Salta a INIT

                                            ;Routine lampeggio giallo
0160  F3           GIALL:  DI               ;Disabilita gli interrupt
0161  3E 00                LD    A,00H      ;Carica A=00
0163  D3 09                OUT   (09H),A    ;Pone A in uscita sulla por-
                                            ;ta D resettando il clock
0165  3E 01                LD    A,01H      ;inizializzazione
0167  32 28 05             LD    (PUM),A    ;orologio notte
016A  3E 12        TEN:    LD    A,12H      ;Carica A =12 (configura-
                                            ;razione G1 - G2)
016C  D3 08                OUT   (08H),A    ;Pone A in uscita sulla
                                            ;porta C
016E  CD 09 04             CALL  LOP        ;Salta alla routine di
                                            ;ritardo
0171  CA 8A 01             JP    Z,ZAC      ;Se (PUM)=0 salta a ZAC
0174  CD 09 04             CALL  LOP        ;Salta alla routine di
                                            ;ritardo
0177  CA 8A 01             JP    Z,ZAC      ;Se (PUM)=0 salta a ZAC
017A  3E 00                LD    A,00H      ;Carica A =00 (configura-
                                            ;zione Spento-Spento)
017C  D3 08                OUT   (08H),A    ;Pone A in uscita sulla
                                            ;porta C
017E  CD 09 04             CALL  LOP        ;Salta alla routine di
                                            ;ritardo
```

Semaforo cibernetico

```
0181   CA 8A 01               JP    Z,ZAC      ;Se (PUM)=0 salta a ZAC
0184   CD 09 04               CALL  LOP        ;Salta alla routine di
                                               ;ritardo
0187   C2 6A 01               JP    NZ,TEN     ;Se(PUM)>0 salta a TEN
018A   FB             ZAC:    EI               ;Abilita le interruzioni
018B   C3 21 01               JP    TAN        ;Ritorna al ciclo prin-
                                               ;cipale

                                               ;Conteggio auto e aggior-
                                               ;namento tempi
0200   F3             CONT:   DI               ;Disabilita gli interrupt
0201   D9                     EX    X          ;Scambia coppie registri
0202   08                     EX    AF,AF'     ;Scambia AF con AF'
0203   78                     LD    A,B        ;Carica B in A (B=numero
                                               ;macchine contate nella
                                               ;configurazione attuale)
0204   91                     SUB   C          ;Sottrae C ad A (C=numero
                                               ;macchine contate nella
                                               ;configurazione attuale
                                               ;nella stessa direzione di B
                                               ;ma nel verso opposto)
0205   F2 0C 02               JP    P,BAN      ;Se il risultato della sot-
                                               ;trazione e' positivo salta
                                               ;a BAN
0208   51                     LD    D,C        ;Carica C in D (poiche' e'
                                               ;maggiore di B)
0209   C3 0D 02               JP    BEN        ;Salta a BEN
020C   50             BAN:    LD    D,B        ;Carica B in D (poiche' e'
                                               ;maggiore di C)
020D   7A             BEN:    LD    A,D        ;Carica D in A
020E   94                     SUB   H          ;Sottrae H ad A (H=numero
                                               ;macchine contate nella
                                               ;configurazione precedente)
020F   FA 33 02               JP    M,ZIN      ;Se il risultato dell sot-
                                               ;trazione e' negativo salta
                                               ;a ZIN
0212   95                     SUB   L          ;Sottrae L ad A (L=margine
                                               ;determinante il cambio dei
                                               ;tempi nella routine di du-
                                               ;rata accensione configura-
                                               ;zioni semaforiche)
0213   FA 2C 02               JP    M,NORM     ;Se il risultato della sot-
                                               ;trazione e' negativo salta
                                               ;a NORM
0216   FD 7E 01               LD    A,(IY+01)  ;Carica A=configurazione suc-
                                               ;cessiva
0219   D6 0C                  SUB   0C         ;Confronta se questa e' 0C
021B   C2 25 02               JP    NZ,MAGG    ;Se <> 0 salta a MAGG
021E   DD 21 07 05    MIN:    LD    IX,0507    ;Inizio stack tempi minori
0222   C3 42 02               JP    CONTR      ;Salta a CONTR
0225   DD 21 0E 05    MAGG:   LD    IX,050E    ;Inizio stack tempi maggiori
0229   C3 42 02               JP    CONTR      ;Salta a CONTR
022C   DD 21 00 05    NORM:   LD    IX,0500    ;Inizio stack tempi uguali
0230   C3 42 02               JP    CONTR      ;Salta a CONTR
0233   85             ZIN:    ADD   A,L        ;Addiziona L ad A (L=margine
                                               ;determinante il cambio dei
                                               ;tempi nella routine di du-
                                               ;rata accensione configura-
                                               ;zioni semaforiche)
0234   F2 2C 02               JP    P,NORM     ;Se il risultato dell'addi-
                                               ;zione e' positivo salta
                                               ;a NORM
0237   FD 7E 01               LD    A,(IY+01)  ;Carica A=config. successiva
023A   D6 0C                  SUB   0C         ;Confronta se questa e' 0C
023C   CA 25 02               JP    Z,MAGG     ;Se =0 salta a MAGG
023F   C3 1E 02               JP    MIN        ;Salta a MIN
0242   62             CONTR:  LD    H,D        ;Carica D in H salvando il
                                               ;numero maggiore di macchi-
                                               ;ne contate nella configu-
                                               ;razione attuale che servi-
                                               ;ra' per la comparazione
                                               ;successiva
0243   06 00                  LD    B,00H      ;Azzeramento dei contatori
0245   48                     LD    C,B        ;macchine B e C
0246   D9                     EX    X          ;Scambia coppie registri
0247   08                     EX    AF,AF'     ;Scambia AF con AF'
0248   FB                     EI               ;Abilita l'interrupt
```

Semaforo cibernetico

```
0249   3E 0C                LD    A,0C       ;Carica A=0C (V1 - R2)
024B   FD A6 01             AND   (IY+01)    ;AND logico tra A e il
                                             ;contenuto di IY+1
024F   CA 34 01             JP    Z,TUN      ;Se=0 salta a TUN
0251   C3 3A 01             JP    INIT       ;Salta a INIT

                                             ;Routine gestione
                                             ;interrupt (testa i 6
                                             ;bit piu' leggeri)
0300   F3          RIVEL:   DI               ;Disabilita gli interrupt
0301   D9                   EX    X          ;Scambia coppie registri
0302   08                   EX    AF,AF'     ;Scambia AF con AF'
0303   2E 05                LD    L,05H      ;Temporiz. V/G-R
0305   DB 09                IN    A,(09H)    ;Pone in A il dato presente
                                             ;sulla porta D
0307   32 24 05             LD    (LUC),A    ;Salva in LUC il dato
                                             ;sulla porta D
030A   CB 47                BIT   0,A        ;Testa il bit 0
030C   CA 2A 03             JP    Z,ZAN      ;Se =0 salta a ZAN
030F   3E 1C                LD    A,1CH      ;Conf.V1/G1-R2
0311   D3 08                OUT   (08),A     ;Pone A in uscita
0313   CD 00 04    STAN:    CALL  DELAY      ;Salta a DELAY
0316   2D                   DEC   L          ;Decrementa L
0317   C2 13 03             JP    NZ,STAN    ;Se L>0 salta a STAN
031A   3E 21                LD    A,21H      ;Carica A=21 configu-
                                             ;razione R1 - V2
031C   2E 8E       SAN:     LD    L,8EH      ;Temporizzazione R - V
031E   D3 08                OUT   (08H),A    ;Pone A in uscita sulla
                                             ;porta C
0320   CD 00 04    SEN:     CALL  DELAY      ;Salta alla routine di
                                             ;ritardo
0323   2D                   DEC   L          ;Decrementa L
0324   C2 20 03             JP    NZ,SEN     ;Se L > 0 salta a SEN
0327   C3 64 03             JP    FEN        ;Salta a FEN
032A   CB 4F       ZAN:     BIT   1,A        ;Testa il bit 1
032C   CA 3F 03             JP    Z,ZUN      ;Se =0 salta a ZUN
032F   3E 23                LD    A,23H      ;Configurazione R1 -V2/G2
0331   D3 08                OUT   (08),A     ;Pone A in uscita porta C
0333   CD 00 04    STIN:    CALL  DELAY      ;Salta a DELAY
0336   2D                   DEC   L          ;Decrementa L
0337   C2 33 03             JP    NZ,STIN    ;Se L>0 salta a STIN
033A   3E 0C                LD    A,0CH      ;Carica A= 0C  (V1 - R2)
033C   C3 1C 03             JP    SAN        ;Salta a SAN
033F   FD 7E 00    ZUN:     LD    A,(IY+00)  ;In A la conf. attuale
0342   E6 24                AND   24         ;AND logico tra A e 24
0344   CA 64 03             JP    Z,FEN      ;Se =0 salta a FEN
                                             ;siamo nella config. R1- R2
0347   FD 7E 00             LD    A,(IY+00)  ;In A la conf. attuale
034A   CB 47                BIT   0,A        ;Testa il bit 0 di A
034C   3A 24 05             LD    A,(LUC)    ;In A il dato presente
                                             ;sulla porta D
034F   CA 5A 03             JP    Z,TES1     ;Se =0 salta a TES1
                                             ;direzione 2
0352   CB 57                BIT   2,A        ;Testa il bit 2 di A
0354   C2 63 03             JP    NZ,TIC     ;Se <> 0 salta a TIC
0357   C3 5F 03             JP    PIP        ;Salta a PIP
035A   CB 67       TES1:    BIT   4,A        ;Testa il bit 4 di A
035C   CA 63 03             JP    Z,TIC      ;Se =0 salta a TIC
035F   04          PIP:     INC   B          ;Incrementa contatore auto
                                             ;in un verso
0360   C3 64 03             JP    FEN        ;Salta a FEN
0363   0C          TIC:     INC   C          ;Incrementa contatore auto
                                             ;nell'altro verso
0364   2E 20       FEN:     LD    L,20H      ;Carica L =20 (ripristina
                                             ;il valore inziale di L)
0366   D9                   EX    X          ;Scambia coppie registri
0367   08                   EX    AF,AF'     ;Scambia AF con AF'
0368   FB                   EI               ;Abilita l'interrupt
0369   ED 4D                RETI             ;Torna dall'interruzione

                                             ;Routine Orologio
0370   DD E5       DAY:     PUSH  IX         ;Salva IX
0372   DD 21 26 05          LD    IX,0526    ;Imposta IX=0526 (PIM)
0376   DD 35 00             DEC   (IX+00)    ;Decrementa (PIM)
0379   C2 89 03             JP    NZ,TT      ;Se <> 0 salta a TT
037C   DD 35 01             DEC   (IX+01)    ;Decrementa (PAM)
037F   C2 89 03             JP    NZ,TT      ;Se <> 0 salta a TT
0382   DD 36 01 E1          LD (IX+01),E1    ;Rimposta il valore di (PAM)
0386   DD 35 02             DEC   (IX+02)    ;Decrementa (PUM)
0389   DD E1       TT:      POP   IX         ;Ripristina il valore di IX
038B   C9                   RET              ;Chiusura routine

                                             ;Genera 0.5 sec. di ritardo
                                             ;e il clock di freq. 2 Hz
0400   3A 25 05    DELAY:   LD    A,(LAC)    ;Carica in A il contenuto
                                             ;della locazione 0525 (sta-
                                             ;to precedente del bit 6)
0403   2F                   CPL              ;Complementa A
0404   D3 09                OUT   (09H),A    ;Pone A in uscita sulla
                                             ;porta D
0406   32 25 05             LD    (LAC),A    ;Salva il contenuto di A
0409   CD F2 F9    LOP:     CALL  BAUD       ;Ritardo di 1.6 ms
040C   15                   DEC   D          ;Decrementa D
040D   C2 09 04             JP    NZ,LOP     ;Se D > 0 torna a LOP
0410   CD 70 03             CALL  DAY        ;Salta a DAY
0413   C9                   RET              ;Chiusura routine

F9F2   C5          BAUD:    PUSH  BC         ;Salva i registri BC
F9F3   ED 46                IM    0          ;Modo 0 di interruzione
F9F5   AE          DIDI:    XOR   (HL)       ;OR esclusivo sulla loca-
                                             ;zione indirizzata da HL
F9F6   0F                   RRC   A          ;Rotazione accumulatore
F9F7   0B                   DEC   BC         ;Decrementa BC
F9F8   CB 78                BIT   7,B        ;Testa il bit piu'pesante
F9FA   28 FB                JR    Z ,DIDI    ;Se il bit=0 va a DIDI
F9FC   C1                   POP   BC         ;Ripristina BC
F9FD   C9                   RET              ;Chiusura routine

                                             ;Abilitazione porta C
0440   3E 0F       PCO:     LD    A,0FH      ;Parola di controllo in A
0442   D3 0A                OUT   (0AH),A    ;Abilitazione porta C
                                             ;modo 0
0444   C9                   RET              ;Chiusura routine

                                             ;Abilitazione porta D
0450   3E 20       PDI:     LD    A,20H      ;Carica in A la parte bassa
0452   D3 0B                OUT   (0BH),A    ;dell'indirizzo
0454   3E FF                LD    A,FFH      ;Parola di controllo in A
0456   D3 0B                OUT   (0BH),A    ;Abilitazione porta D
                                             ;modo 3
0458   3E 3F                LD    A,3FH      ;Linee di I/O= entrata
045A   D3 0B                OUT   (0BH),A    ;6 bit piu' leggeri
045C   3E B7                LD    A,B7H      ;Parola di controllo
045E   D3 0B                OUT   (0BH),A    ;interruzioni abilitate
                                             ;attivo alto
0460   3E C0                LD    A,C0H      ;Maschera: cosidera i
0462   D3 0B                OUT   (0BH),A    ;bit 0,1,2,3,4,5
0464   C9                   RET              ;Chiusura routine
```

ICON			MINS			NORMS			MAGGS		
IND.	COD.	CONFIGURAZIONI	IND.	COD.	SEC.	IND.	COD.	SEC.	IND.	COD.	SEC.
0515	21	Ros1-Ver2	0507	16	9	0500	36	22.08	050E	56	35.17
0516	23	Ros1-Ver2/Gia2	0508	0B	4.5	0501	0E	5.72	050F	13	7.77
0517	24	Ros1-Ros2	0509	03	1.22	0502	05	2.04	0510	05	1.22
0518	0C	Ver1-Ros2	050A	56	35.17	0503	36	22.08	0511	16	9
0519	1C	Ver1/Gia1-Ros2	050B	13	7.77	0504	0E	5.72	0512	0B	4.5
051A	24	Ros1-Ros2	050C	03	1.22	0505	05	2.04	0513	03	1.22
051B	00	---------									

Tabella 1 - Tabella delle configurazioni e dei tempi.

```
Routines          Labels                        Locazioni di mem.

0100: START       0121: TAN      031C: SAN      0524: LUC
0160: GIALL       0134: TUN      0320: SEN      0525: LAC
0200: CONT        013A: INIT     032A: ZAN      0526: PIM
0300: RIVEL       0147: TIN      0333: STIN     0527: PAM
0370: DAY         016A: TEN      033F: ZUN      0528: PUM
0400: DELAY       018A: ZAC      035A: TES1     0540: SP
0440: PCO         020C: BAN      035F: PIP
0450: PDI         020D: BEN      0363: TIC      gestione
F9F2: BAUD        021E: MIN      0364: FEN      interrupt
                  0225: MAGG     0389: TT       0520: 00
                  022C: NORM     0409: LOP      0521: 03
                  0233: ZIN      0427: STUR
                  0242: CONTR    F9F5: DIDI
                  0313: STAN
```

Tabella 2 - Tabella dei simboli.

Vediamo i parametri che interessano tale progetto:

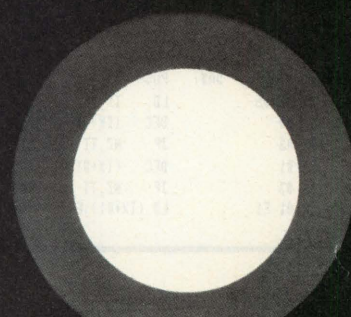
fo = 4 KHz frequenza centrale
Av = 50 amplif. di tensione
Q = 10 fattore di qualità
B = F0/Q = 400 Hz ampiezza banda
Rm1= Rm1' = 600 Ohm microfoni
W0 = 2ττF
Rm1/ Rm1' = 300 ohm
C1 = Q/Rm1 ★WO★AV = 0.022 μF
C1 = C3
R25= Q (C1+C3)/W0 ★C1★ C3 = Q/ττf0 C 300 Kohm
Rp = Rm1//R27 = 1/Wo★1★2★R25★C1★C3 =7,6 ohm
R27= Rm1-Rp / Rm1★Rp = 8 ohm

La resistenza R27, è un trimmer da 1 Kohm per poter variare la fo e centrare in fase di taratura la banda della sirena (vedi "La progettazione dei filtri attivi", ed. Jackson).
Avendo il segnale dei microfoni un'ampiezza massima di qualche decina di mV, il segnale amplificato che si ottiene in uscita del filtro non è adeguato a livelli logici digitali. Abbiamo pensato quindi di raddrizzare e stabilizzare il segnale con D25 e C5 per poi compararlo con un livello di riferimento.
La resistenza R31 in parallelo al condensatore C5, è posta per la scarica dello stesso. Il comparatore ha dunque al suo ingresso non invertente il segnale filtrato, amplificato e raddrizzato e al suo ingresso invertente il valore di riferimento preso dal centrale del trimmer multigiri R33.
All'uscita del comparatore è posto il diodo zener da 5,1 V Dz 5 e la resistenza di caduta R29. Avremo quindi ai capi dello zener 5V (stato logico 1), se il segnale avvertito dalle capsule microfoniche è di circa 4 KHz, altrimenti avremo tensione nulla (stato logico 0). Come si diceva, non è solo il suono della sirena ad avere frequenza pari a 4 KHz. Può accadere infatti che i rivelatori acustici generino l'impulso anche in corrispondenza di suoni occasionali aventi tale frequenza.
Abbiamo dunque utilizzato la durata del segnale come parametro, determinando il tempo necessario a stabilire la natura del suono in almeno 4 secondi. Abbiamo inserito un chip 7490 in grado di contare gli stati del clock (2 Hz) e di emettere dopo otto impulsi il comando di interrupt.
Quindi il catodo dello zener Dz5 è stato collegato all'ingresso di un buffer, e l'uscita di questi è stata connessa al pin di reset Mr1 del contatore. Dunque il 7490 sarà resettato se ai capi dello zener la tensione è nulla mentre, se la tensione è 5V, inizierà a contare e porrà S1 allo stato alto dopo 8 impulsi di clock.
Nel caso in cui, durante il conteggio degli 8 impulsi, Mr1 si porti a livello alto (cioè il suono è occasionale), il conteggio viene interrotto, il contatore si resetta e S1 rimane a livello basso. Le uscite dei due contatori S1, S2 sono connesse con i due pin PD0 e PD1 della P10 Z80 genera l'eventuale interrupt.

Blocco G: routine interrupt sirena

Facciamo riferimento ora ad una parte dell'articolo pubblicato nel numero precedente. Infatti per ragioni di praticità si è realizzata un'unica routine di gestione interrupt, la routine *RIVEL* (blocco D) che provvede a gestire sia i comandi inviati dal blocco C che quelli dal blocco F. Per quanto riguarda i comandi provenienti da quest'ultimo blocco, la routine procede testando i bit PD0 e PD1 della porta D.
Se trova uno di questi alto, indipendentemente dai problemi di conteggio, attiva il verde nella direzione corrispondente.

Cablaggio e taratura

Come sempre, una prima realizzazione è stata effettuata su

breadboard. Dopo aver verificato la funzionalità del montaggio, abbiamo cablato il tutto su basetta eurocard prestampata a piazzole. I collegamenti sono stati effettuati con spezzoni di filo. Da questo tipo di montaggio deriva il particolare disegno dello stampato presentato. nel mobile realizzato trova alloggio il nanocomputer NBZ80-S, i cui collegamenti con il circuito stampato avvengono mediante gli zoccoli del cavo flat a 18 capi (figura 11). Il collegamento tra lo stampato ed i componenti montati sul pannello, avviene tramite il connettore femmina a 21 poli, nel quale si inserisce la basetta eurocard, che viene così sorretta anche fisicamente.

Descriviamo infine la procedura di taratura delle parti che lo necessitano.

Nel blocco C (rilevatore di passaggio vetture, figura 6), abbiamo direzionato FR1 sul LED D21 e agendo sul trimmer R17 abbiamo portato la tensione sul terminale invertente dello OP AMP a 3 V. Operando poi sul trimmer R21, abbiamo posto sul terminale non invertente una tensione di 2,5 V.

Abbiamo ripetuto tale operazione nei rimanenti tre versi con R18-R22, R19-R23, e R20-R24.

Nel blocco F (rivelatore vetture con sirena, figura 7), abbiamo posto nelle vicinanze di uno dei microfoni una sorgente sonora (registratore amplificatore altoparlante) con il segnale della sirena inciso in precedenza su cassetta.

Agendo sul trimmer R27 abbiamo cercato, centrando la banda, di ottenere il massimo livello di tensione ai capi di C5. Nel nostro caso sono stati ottenuti valori di circa 2 V, ma le variabili di questa fase (distanza dell'altoparlante dai microfoni, tipi di microfoni, tolleranza dei condensatori) rendono impossibile una standardizzazione. Il tutto non crea comunque eccessivi problemi, in quanto agendo su R33 è possibile adeguare il livello di riferimento del comparatore, che nel nostro caso è stato posto a 1,5 V.

La stessa procedura è stata rivolta alla direzione opposta con R28 e R34.

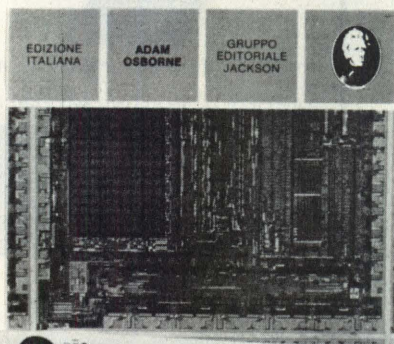
Le misure sono state effettuate con un multimetro digitale a 3 1/2 cifre.

Conclusioni ed osservazioni

Per un'applicazione concreta della nostra ricerca, pensiamo sia indispensabile l'uso di un diverso e più sicuro sistema di acquisizione dati. Vogliamo dire che le vetture che lo necessitano (polizia, autoambulanza ecc.), potrebbero fornire dei segnali appropriati dalle trasmittenti in dotazione. Anche la rivelazione del numero delle vetture transitanti potrebbe avvenire diversamente, sfruttando fenomeni magnetici dovuti al passaggio di corpi metallici, o ancora con trasduttori ottici posti diversamente. Ma questo esula dal nostro studio e dalle nostre intenzioni. Infatti l'osservazione dà cui è partito il nostro lavoro, è che molto spesso la creatività dell'uomo, nel nostro caso un vigile, viene frustata da compiti meccanici e ripetitivi cui invece la macchina può e deve ottemperare efficacemente. Così da non vedere trasformato un professionista della viabilità, cui compete la risoluzione di problemi ben più complessi, in un mero manovratore manuale di un semaforo "poco" automatico.

Bibliografia

1) Millman, Jacob Halkias, Christos C. Integrated electronics, analog and digital circuits and sistems, Mac Graw-Hill, New York, 1972 (tr. it. C. Naldi e G. Conte, Microelettronica, Boringhieri, Torino, 1978, (1 ristampa 1979)).
2) Berlin, Haward M. La progettazione dei filtri attivi, esperimenti, Jackson Editrice, Milano, 1979.
3) Nichols E.A. Nichols J.C. Rony P.R. Il Nanobook Z-80, Vol. 1, Tecniche di programmazione, Jackson Editrice, Milano, 1979.
4) Nichols E.A. Nichols J.C. Rony P.R. Il Nanobook Z-80, Vol. 3, Tecniche di interfacciamento, Jackson Editrice, Milano, 1979.
5) Fairchild, TTL Data Book
6) National, Linear Data Book.

8080 PROGRAMMAZIONE E PROGETTAZIONE LOGICA

Il libro descrive l'implementazione della logica sequenziale e combinatoria con l'uso del linguaggio assembly all'interno di un sistema a microcomputer basato sull'8080.

Lo scopo è quello di insegnare ai progettisti logici come eseguire in modo nuovo un vecchio lavoro mediante la creazione di programmi e ai programmatori come la programmazione abbia trovato uno scopo nuovo nel progetto logico.

I concetti tradizionali di programmazione in linguaggio assembly non sono utili né attinenti per usare i microprocessori in applicazioni logiche digitali: l'uso delle istruzioni in linguaggio assembly per simulare il packages digitale è in tutti i casi errato.

Il libro chiarifica questi concetti per prima cosa simulando sequenze logiche digitali, poi illustrando alcune efficienti soluzioni per spiegare l'uso corretto dei microcomputer. Un capitolo, infine, contiene il set completo di istruzioni dell'8080.

Sommario

Introduzione - Linguaggio assembly e logica digitale - Una simulazione diretta della logica digitale - Un semplice programma - Prospettiva del programmatore - Set di istruzioni - Alcune subroutine impiegate comunemente - Codici di caratteri ASCII.

Pagg. 296
Prezzo L. 19.000

Formato 14,5 x 21
Codice 325P

Per ordinare il volume utilizzare l'apposito tagliando inserito in fondo alla rivista.

GRUPPO EDITORIALE JACKSON
Divisione Libri

La biblioteca di Bit

a cura di **A. Cavalcoli**

PRESENTAZIONE

Questo mese la rubrica vi presenta un testo che forse era meglio vedere in libreria molto tempo fa.
Ma non è colpa di nessuno; e poi, anche ora, nulla perde della sua importanza come strumento informativo e di lavoro.
Si tratta della Guida all'Uso dell'Apple II.
E tanto per non perdere di vista l'aspetto "tempo libero" ed intrattenimento" dei PC, perché non dare un'occhiata ad un libro di giochi? Ed ecco a voi "Giocare con il BASIC".

Apple II, Guida all'Uso
di L. Poole, M. McNiff e S. Cook
Gruppo Editoriale Jackson

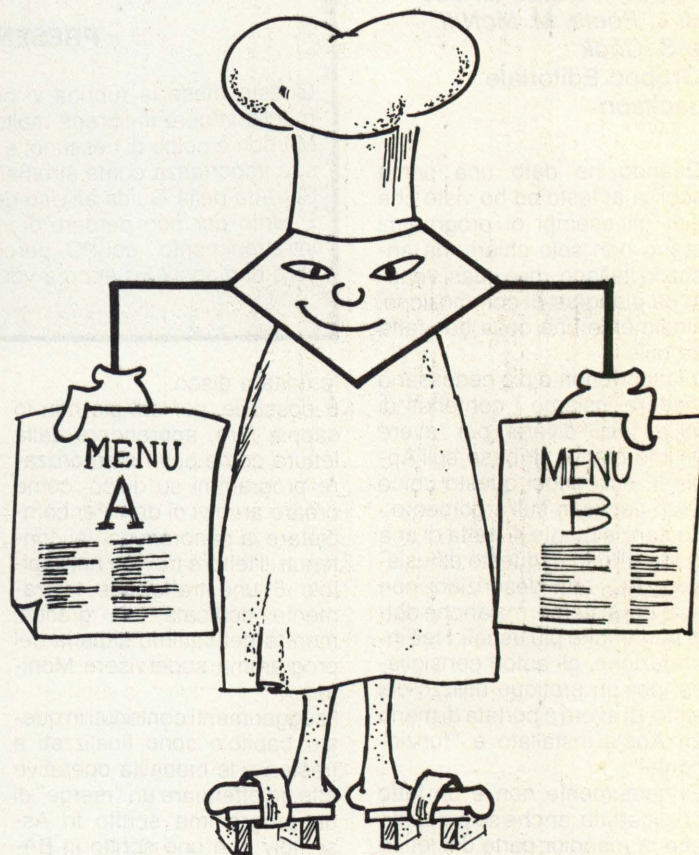
Quando ho dato una prima scorsa al testo ed ho visto che tutti gli esempi di programmi erano non solo chiari, ma anche in italiano, mi è quasi venuto da piangere di commozione. Finalmente una cosa ben fatta ed utile.
Infatti ora non è più necessario mettere insieme i contenuti di mille testi diversi per avere un'informativa di base sull'Apple II, essendoci questo unico testo Jackson sull'argomento.
Sostanzialmente si tratta di una guida all'uso di questo diffusissimo PC, con descrizioni non solo del sistema, ma anche delle periferiche più usuali. Nell'introduzione, gli autori consigliano, per un proficuo utilizzo del testo, di avere a portata di mano un Apple installato e "funzionante".
Evidentemente non è un fatto obbligatorio, anche se presumo che la maggior parte dei lettori sarà costituita da attuali possessori della macchina in questione, che non hanno bisogno di sapere come si installa un Apple perché l'hanno già fatto, oppure perché qualcuno l'ha fatto per loro.
Quindi, sia ben chiaro, questo è un libro che vi dice come usare la macchina. Dopo una descrizione dei vari elementi base del sistema, si passa alla parte pratica, alla spiegazione di come usare la macchina con una prima serie di programmi dimostrativi pronti.
Una volta acquisita la macchina nella sua operatività base, i capitoli successivi sono dedicati allo studio, sempre sperimentale, del BASIC: ma attenzione, si tratta del BASIC Apple, cioè dell'Integer BASIC e dell'Applesoft. Personalmente trovo particolarmente utile il capitolo 5, interamente dedicato alle unità a disco.
È possibile, per chi già non lo sappia fare, apprendere dalla lettura come poter memorizzare programmi su disco, come creare archivi di dati. Per completare la panoramica dei contenuti, il lettore troverà nel capitolo 6 una trattazione interamente dedicata alla grafica, mentre nel settimo si parla del programma supervisore Monitor.
I suggerimenti contenuti in questo capitolo sono finalizzati a trasferire le modalità operative atte ad effettuare un "merge" di un programma scritto in Assembly con uno scritto in BASIC.
Infine, tabelle ed appendici varie.
Questo libro è il classico "must", vale a dire un obbligo per gli attuali possessori di Apple, ma anche un innocuo investimento a livello informativo per futuri potenziali utenti Apple indecisi.

Giocare con il BASIC
di R. Mateosian
Gruppo Editoriale Jackson

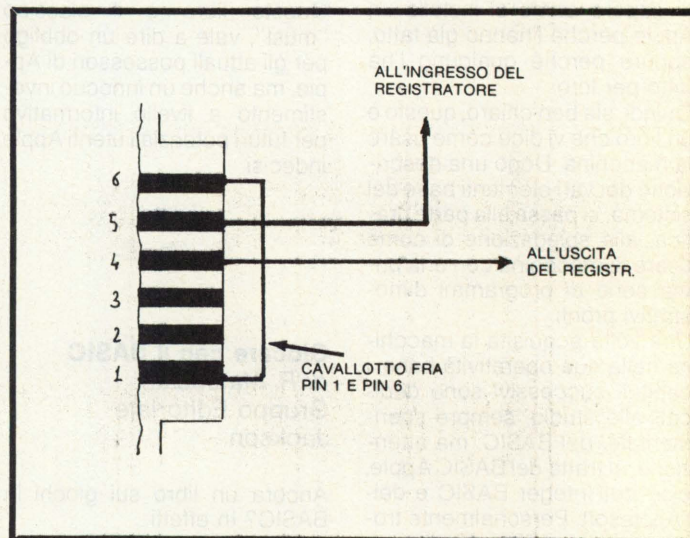
Ancora un libro sui giochi in BASIC? In effetti...
Quindi, lo gettiamo via?
Prima vedere, poi decidere.
Di libri sui giochi in BASIC ne sono stati scritti molti, alcuni interessanti, decorosi, altri banali, oppure orrendi nella loro eccezionale inutilità. Ricordo uno dei primi libri sul BASIC che avevo letto: si trattava di BASIC Computer Games, uno dei primi del suo genere, appunto.
Non ne ero rimasto molto eccitato, anche perché si trattava di software "pronto", da consumare, senza alcuna possibilità di incremento culturale. L'elencazione di giochi pronti è inutile, mentre ben diversamente vanno considerati libri in cui si fa vedere come nasce un programma, in cui viene esplicitato l'algoritmo.
In tal caso, diventa possibile una fase di studio, per una personale serie di iniziative autonome sul proprio PC, indipendentemente da quello utilizzato dall'autore.
In questo libro, la parte culturale non va disattesa, ed è questo che qualifica il testo, almeno una sua buona parte.
Dire che un approccio alternativo al BASIC, rispetto a quello classico di un duro studio, è quello consistente nel giocare, nel fare giochi da cui trarre una immediata e fallace gratificazione, per me è completamente falso.
Anche perché i Game rappresentano un settore applicativo praticamente autonomo, con un loro mercato ricco e dinamico.
Di conseguenza è cosa seria ed impegnativa, che non si impara, volendo fare buone cose, leggendo o copiando idee da un libro.
Nei libri di giochi, e quindi anche in questo, non troverete nuovi giochi, ma la ripetizione, la rielaborazione di vecchi giochi; e se novità ci sono, non rappresentano certo un grosso salto di qualità, perché se così fosse, l'autore non li avrebbe descritti nel libro, ma se li sarebbe allegramente venduti a parte.
Quindi, tanto per ripetere il concetto, l'importanza ed il significato di un decoroso libro sui giochi, quale questo è, risiede essenzialmente nella chiara e didattica esposizione del metodo, dall'algoritmo con cui fare. Starà poi al lettore associare ai vari metodi le sue idee, la sua fantasia per dar vita veramente a cose nuove ed interessanti, professionalmente valide, con una loro, perché no, vendibilità.
Ho tenuto ad essere chiaro, perché non è certo mia intenzione deludere quanti vorranno leggere questo libro.
Anche, ho voluto sottolineare alcuni aspetti importanti che spesso sfuggono al lettore poco attento, che potrebbe essere portato a snobbare alcune offerte tecnico-culturali (questo libro) a causa del titolo palesemente "ludico".
Bene. Adesso sta a voi decidere se vi interessa o meno.
Una cosa è certa: se intendete un domani diventare ricchissimi creando e vendendo giochi, penso che siate obbligati ad acquistare questo "Giocare con il BASIC". ■

Il ricettario

Arriva l'hardware

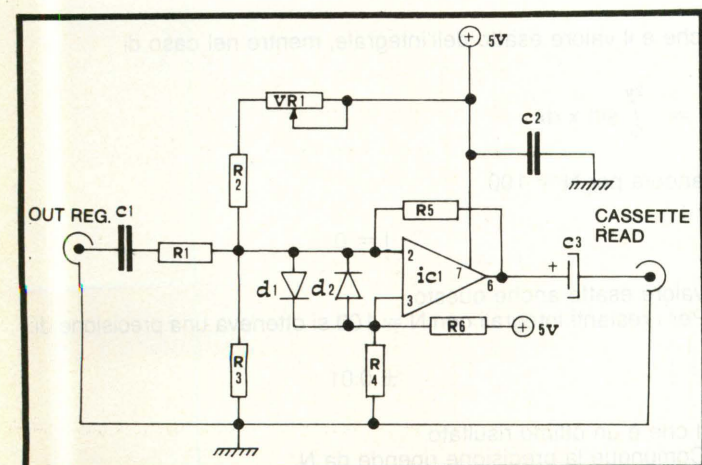
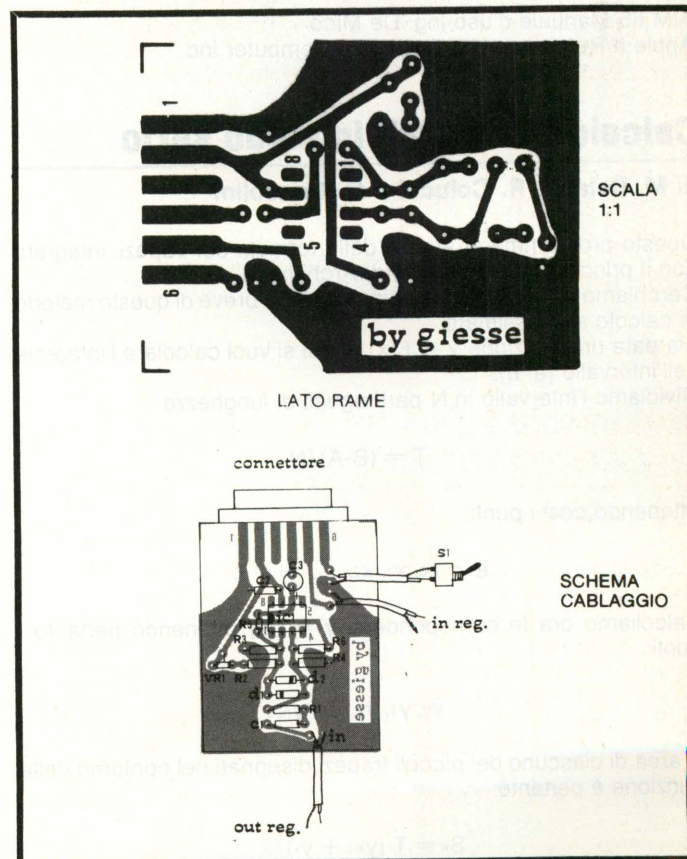
Decisamente qualcuno ci legge nel pensiero: giuro che *prima* che lanciassi l'appello (quasi disperato) all'invio di qualche espediente e/o dispositivetto materiale - non solo a contrastare l'imperialismo trionfante del sofficiume, quanto a render la vita meno amara agli stessi softwaristi, senza dover "*comprasse la chitarra*" di petrolinian-manfrediana memoria - eccoti che arriva il circuitetto di Giuseppe Giovanni Selgi. Costui non aspira certo al Nobel dell'elettronica, però risolve concretamente i problemini del non facile matrimonio tra VIC 20 e registratori del tipo volgaris. E chi sa che l'ideuzza, in sè banale, di innalzare il livello deboluccio e ondaquadrarlo a dovere non sia riciclabile - mutatis paucis mutandis - in altri ambienti signal-anemici.
Al buon Selgi dunque l'onore di aprire come primo piatto il menu del mese. Seguono stavolta, per contentar li patiti della Mathematica Scientia, temi di caledo. Apre la schiera di questo - un poco irritabile - genus un trio di goliardi dalla scarsella alquanto verde e ricca, al più, di ragnatele. Fanno la consueta questua dei clerici vaganti abbinandola a più moderne contestazioni: dicono infatti, i pedantoni, che gl'integrali calcolati a Montecarlo son troppo ... aleatori e lenti per sovrammercato. Accontentiamoli, anche perchè tutti i torti non li hanno.
Proseguendo sul serioso c'è poi un quasi-professoral trattato d'interpolazione lagrangiana (però serve, eccome!).
Dulcis in fundo chiude però l'inclita miniprocessione una routine assai praticosa, che surroga il printusing anche sui BASIC più spartani.
Speriamo d'aver accontentato tutti.
A questo punto non resta che ricordare a quanti vogliono il compenso standard di lire trentamila, avendo pubblicato una ricetta, di inviare, all'attenzione di G. Giaccaglini, i loro completi dati anagrafici unitamente al codice fiscale.
Quelli che fossero iscritti all'IVA (non Zanicchi... dio quant'è scema questa) inviino direttamente la loro fatturina all'amministrazione Jackson.

Matrimonio solido tra VIC 20 e registratore comune

di G. G. Belgi

Come tutti i VIC-omani ben sanno sul VIC 20 c'è un connettore per collegare, come memoria di massa il datacassette VIC - C2N, un registratore audio carrozzato Commodore. Ma forse non tutti conoscono i problemi che saltano fuori quando si tenta di collegarvi un registratore comune, anche se la cosa sembrerebbe di una semplicità mostruosa (vedi fig. 1).
Anch'io lo feci a suo tempo.
Scrivo un programmino di prova e salvatolo tramite l'arcinota SAVE PIPPO, verifico che la registrazione è avvenuta, VERIFY PIPPO, ed il VIC risponde OK. Quindi, convinto dell'affidabilità del tutto, mi metto a digitare programmi da "incassettare", dopo aver controllato che girassero.

Il giorno dopo, acceso il VIC e digitato il fatidico LOAD PIPPO aspettavo con impazienza il caricamento di una utility: ... OUT OF MEMORY... fu la risposta del sistema operativo! Trascorsi 98,56 minuti nell'angoscia più nera, ritornai in me speranzoso che il tutto dipendesse da crisi passeggera e incrociate le dita, riprovai.... e dopo 826 tentativi andati a vuoto mi resi conto che ripescare un programma salvato sul nastro di un normale registratore (i VIChinghi possono testimoniare) dipende solo dai rapporti personali con la dea bendata, nel mio caso praticamente inesistente.
Aiutati che la dea t'aiuta, ho pensato di simularla artificialmente. Ad harware.
Aperto il VIC cominciai a smanettar dentro per vedere dove andava a finire il segnale proveniente dal registratore: va al pin 40 (CA1) di una VIA (Versatile Interface Adaptator) di tipo 6522.
Spulciando i data scheets dell'integrato e confrontando con altri circuiti tratti da vari libri (vedi riferimenti bibliografici) conclusi che la lettura anomala avviene in quanto l'uscita del registratore non riesce a fornire un livello di tensione atto a pilotare correttamente il 6522.
Ancora: essendo il segnale in uscita dal registratore non sufficientemente squadrato anche questo rende difficoltosa la lettura.
Qualche giorno dopo vedeva la luce il circuito che mi appresto a descrivere (*lo battezziamo PIPPO - È - SALVO?* Ndr).

Descrizione del circuito

Il nucleo centrale è un economico op-amp nA 741 (vedi figura 2) cui è collegata una manciata di vulgaris componentaglia.
Il circuito non è altro che un amplificatore in tensione ad alto guadagno che provvede ad innalzare, a valori compatibili con la logica del 6522, il segnale d'uscita del registratore.
Tale segnale, infatti, è collegato al pin 2 (inverting input) del 741 il quale è polarizzato al pin 3 (non inverting input) in modo da permetterne il funzionamento in singola alimentazione (cioè non simmetrica rispetto massa) a + 5V la quale è disponibile al connettore del VIC stesso.
La resistenza R5 insieme alla R1 stabiliscono l'amplificazione del circuito (circa 1400).
I due diodi all'ingresso (D1 e D2) tosano il segnale d'ingresso (clipping) onde renderlo simile ad un'onda quadra (trapezoidale) e ciò per renderlo più appetibile al 6522.
L'uscita del 741 (pin 6) tramite un condensatore (C3) va al connettore tramite il quale viene collegato all'ingresso CASSETTE READ del VIC.
Il trimmer VR1 permette di tarare con precisione il funzionamento del circuito.

Montaggio

La costruzione del circuito è semplicissima, in particolar modo facendo uso del circuito stampato (vedi figura 3), comunque può andar bene anche una piastra millefori a passo integrati dove i componenti verranno collegati inferiormente tramite spezzoni di filo elettrico.
Lo stampato può essere autocostruito usando un supporto in vetronite ramata e facendo uso di pads trasferibili (Mecanorma, R41, ecc.) e poi inciso con le apposite soluzioni esistenti in commercio (Percloruro ferrico e simili).
Per il montaggio si consiglia di iniziare saldando il connettore sull'apposito pettine e poi proseguire con gli altri componenti (si raccomanda vivamente l'uso di un zoccolo per il 741).
Dopo il montaggio delle parti si controlli attentamente di non aver commesso errori aiutandosi con lo schema di cablaggio di figura 4.
Appena sicuri che tutto è in ordine si può collegare il circuito al pettine posteriore del VIC, il quale si incaricherà di fornirgli la dovuta alimentazione.
Ad interfaccia inserita e VIC acceso si predisponga il tester per una portata di 10 V fondo scala e si colleghi il puntale negativo a massa e quello positivo al pin 2 del 741, smanettando quindi (tramite cacciavitino) il trimmer VR1 fino a leggere sul tester una tensione di 2.5 V.
Fatto ciò anche la taratura è conclusa e non ci resta altro che collegarla al registratore, tramite cavetti schermati per BF.
Predisponiamo adesso il controllo di volume del registratore a 3/4 della sua corsa massima e proviamo a scrivere e a leggere tramite il nostro VIC: normalmente sin dal primo tentativo tutto va bene e si riesce facilmente a caricare i programmi da nastro.
Può però capitare che la lettura non avviene o non è perfetta (LOAD ERROR) ciò è dovuto alla posizione del controllo di volume che non è ottimale.
In tal caso provare ad alzare oppure ad abbassare il volume riprovando fino a trovare la posizione opportuna (ci si riesce facilmente dopo pochi tentativi) segnando poi tale posizione in modo da poterci ritornare facilmente.
Detta interfaccia sul mio VIC è montata già da parecchio tempo senza mai darmi nessun problema, funzionando con prestazioni veramente brillanti.

```
COMPONENTI

R1 - 3,9K   1/4W
R2 - 2,2K   1/4W
R3 - 4,7K   1/4W
R4 - 4,7K   1/4W
R5 - 5,6K   1/4W
R6 - 4,7K   1/4W
C1 - 0,2 F
C2 - 0,1 F
C3 - 5 F 15VL elettrolitico
d1 - 1N 914
d2 - 1N 914
VR1- 4,7K trimmer lineare
ic1- nA 741
s1 - interruttore

Connettore a pettine
femmina 6 poli.
```

Bibliografia

HOWARD M. BERLIN
La progettazione dei circuiti amplificatori operazionali ED. JACKSON
AIM 65 Manuale d'uso ing. De Mico
Apple II Reference manual Apple Computer inc.

Calcolo d'integrali in modo serio

di **M. Belardi, R. Colucci e G. Consolini**

Questo programma si avvale della formula dei trapezi, integrata con il principio di esaustione di Archimede.
Cerchiamo ora di dare una spiegazione in breve di questo metodo di calcolo approssimato.
Sia data una funzione $y = f(x)$ di cui si vuol calcolare l'integrale nell'intervallo (a, b).
Dividiamo l'intervallo in N parti uguali di lunghezza:

$$T = (B-A)/N$$

ottenendo così i punti:

$$a = x_0, x_1, x_2, \dots, x_n = b$$

Calcoliamo ora le corrispondenti $y = f(x)$ ottenendo pertanto i punti:

$$y_0, y_1, y_2, \dots, y_n.$$

L'area di ciascuno dei piccoli trapezi disegnati nel contorno della funzione è pertanto:

$$S_k = T\,(y_{k-1} + y_k)/2$$

pertanto l'integrale essendo la somma di tutte le aree dei piccoli trapezi avremo che risulterà essere pertanto:

$$I = \sum_1^n S_k = T\,(y_0 + y_n + 2y_1 + 2y_2 + \dots + 2y_{n-1})/2 \quad 1)$$

Veniamo ora ad una succinta descrizione del programma. Nella prima parte si definiscono le variabili A, ossia il limite inferiore di integrazione, B cioè il limite superiore, mentre N è il numero di parti in cui si suddivide l'intervallo.
Dal passo 40 al passo 100 si ha un loop allo scopo di calcolare la somma:

$$H = 2y_1 + 2y_2 + \dots + 2y_{n-1}$$

Quindi dal passo 110 al passo 140 viene calcolata l'intera somma che si trova nella parentesi della 1) sommando ad H il valore f(a) ed il valore f(b).
Al passo 150 viene calcolata proprio la 1). Nel passo 1000 utilizzato come subroutine deve essere programmata la funzione di cui se ne vuole calcolare l'integrale, e la funzione deve essere programmata nella forma

$$y = f(x),$$

cioè ad esempio se si vuole calcolare

$$I = \int_b^a (2x + 3)\,dx$$

si porrà

1000 Y = 2X + 3

Il tempo medio di esecuzione del programma per N = 100 è di circa 60 sec, mentre per N = 100 è di circa 10 minuti.
Per quanto riguarda la precisione bisogna dire che in tal senso si è ottenuto realmente un buon valore basta pensare che nel caso di

$$I = \int_0^3 2x\,dx$$

abbiamo ottenuto per N = 100

$$I = 9$$

che è il valore esatto dell'integrale, mentre nel caso di

$$I = \int_0^{2\pi} \sin x\,dx$$

ancora per N = 100

$$I = 0$$

valore esatto anche questo.
Per i restanti integrali con N = 100 si otteneva una precisione di:

$$\pm 0.01$$

il che è un ottimo risultato.
Comunque la precisione ripende da N.

LISTING

```
   5  PRINT "INTEGRALI DEFINITI"
  10  H = 0 : Z = 0
  20  INPUT "A=", A, "B=", B, "N=", N
  30  T = (B - A)/N
  40  FOR K = 1 TO N-1
  50  C = K*T
  60  X = A + C
  70  GOSUB 1000
  80  Z = 2*Y
  90  H = H + Z
 100  NEXT K
 110  X = A : GOSUB 1000
 120  H = H + Y
 130  X = B : GOSUB 1000
 140  H = H + Y
 150  D = (T/2)*H
 160  PRINT "INT.=" ; RND (D,-4)
 170  GOTO 10
1000  ......... (funzione da programmare Y =)
1100  RETURN
```

Riportiamo il listing scritto a macchina da scrivere, scusandoci di non possedere la stampante dell'"FX 702 P", essendo membri della confraternita S.S.S." (studenti senza soldi) non la possediamo.

Microbibliografia

N.S. Bachvalov "METODI NUMERICI" Ed. MIR Editori Riuniti

N.S. Piskunov "CALCOLO DIFFERENZIALE ED INTEGRALE I" Ed. MIR Editori Riuniti)

Interpolare con Lagrange

di S. Mancin

La formula di Lagrange permette, data una funzione empirica, di cui siano noti valori in un numero finito m, di interpolarla con un polinomio di grado n = m—1.
Il tutto nasce dalla formula dell'interpolazione lineare che fornisce i valori y = P1 (x) a partire dai punti (x0, y0) e (x1, y1):

$$[1]\ P1(X) = (X - X1) / (X0 - X1) * Y0 + (X - X0) / (X1 - X0) * Y1$$

dove si osserva che il coefficiente di Y0 (risp. Y1) è un polinomio di primo grado che vale 1 per X = X0 (risp. per X = X1) e vale 0 per X = X1 (risp. per X = X0), onde P1 vale y0 (risp. y1) per X = X0 (risp. X1).
Generalizzando, si possono sommare m addendi per avere un polinomio Pn (X) di grado n:

$$[2]\ Pn(X) = LO(X) * Y0 + L1\ (X) * Y1 + ... + Ln\ (X) * Yn =$$

$$= \sum_{i=}^{n} Li(X) * Yi$$

con LO(X), L1(X), ..., Ln(X) polinomi di grado n tali che Li(Xi) = 1 e Li (Xj) = O con $i \neq j$. Con tale scelta dei polinomi Li(X) (i = 0, 1, ..., n) detti polinomi di Lagrange, risulta:

$$[3]\ Pn(Xi) = Li(Xi) * Yi = Yi.$$

Siccome il generico polinomio di Lagrange (Li(X) deve annullarsi nei punti X0, X1, ..., X(i—1), X(i+1), ..., Xn, può essere scritto.

$$[4]\ Li(X) = K(X-X0)\ (X-X1)\ ...(X-X(i-1))\ (X-X(i+1))...\ ...\ (X-Xn)$$

inoltre, essendo Li(Xi) = 1, si ottiene

$$[5]\ K = 1\ /\ [(Xi-X0)\ ...\ (Xi-X(i-1))\ (Xi-X\ (i+1))\ ...\ ...\ (Xi-Xn)]$$

e quindi sostituendo la [5] nella [4]:

$$[6]\ Li\ (X) = [(X-X0)\ (X-X1)\ ...\ (X-X\ (i-1))\ (X-X\ (i+1))\ ...\ ...\ (X-Xn)]\ /[(Xi-X0)\ (Xi-X1)\ ...\ (Xi\text{-}X(i-1))\ (Xi-X\ (i+1))\ ...\ (Xi-Xn)] =$$

$$= \prod_{\substack{j=0 \\ j\neq}}^{n} (X-Xj)\ /\ (Xi-Xj)$$

Che sostituito nella [2] ci permette di ottenere la celebre formula di Lagrange:

$$[7]\ Pn(X) = \sum_{i=0}^{n} [\prod_{\substack{j=0 \\ j\neq i}}^{n} (X-Xj)\ /\ (Xi-Xj)] * Yi.$$

Il programma

Date le coppie di valori (X0, Y0), (X1, Y1), ..., (Xn Yn), e scelto il punto di interpolazione XS = Xi (I = 0, 1, ..., n), si vuole determinare il valore YS = YS(XS) del polinomio interpolatore. Per la [7]:

$$[8]\ YS = \sum_{i=0}^{n} [\prod_{\substack{j=0 \\ j\neq}}^{n} (XS-Xj)\ /\ (Xi-Xj)] * Yi$$

che al fine di abbreviare il calcolo, si può scrivere:

$$[9]\ YS = \sum_{i=0}^{n} \{[C/\ (XS-Xi)]\ /\ \prod_{\substack{j=0 \\ j\neq 1}}^{n} (Xi-Xj) * Yj\}$$

con:

$$[10]\ C = \prod_{j=0}^{n} (XS-Xj)$$

C è comune a tutti gli addendi della [9] e, posto inizialmente uguale ad 1, è generato dal processo ricorrente

$$C(0) = 1 * (XS-X0)$$
$$C(1) = C(0) * (XS-X1)$$
$$.................$$
$$C(n) = C = (n-1) * (XS-Xn)$$

dove gli indici (0), (1), ..., (n) sono i valori successivamente assunti da j. Il generico addendo della [9]

$$[11]\ T(i) = [C*Yi/(XS-Xi)]/ \prod_{\substack{j=0 \\ j\neq 1}}^{n} (Xi-Xy)$$

$$\text{con } i = 0,\ 1,\ ...,\ n$$

e calcolato, a partire da

$$[12]\ T(i,\ 0) = C*\ Yi/(XS-Xi)$$

tramite il seguente processo ricorrente

T (i, 0)
T (i, 1) = T (i, 0)/(Xi—X0)

....

$T(i,i) = T(i,i-1)/(Xi-X(i-1))$
$T(i,i+1) = T(i,i)/(Xi-X(i+1))$

....

$T(i,n) = T(i) = T(i, n-1)/(Xi-Xn)$.

Eseguendo quindi la somma degli n+1 addendi, si ha in uscita il valore del polinomio in XS:

$$[13]\ YS = T(0) + T(1) + ... + T(n).$$

```
 2        PROGRAM IMA1
 3 C
 4 C      FORMULA DI INTERPOLAZIONE DI LAGRANGE
 5 C      SEQUENZA COLLOQUIALE DI INTRODUZIONE DATI
 6 C
 7        DIMENSION X(100),Y(100)
 8        DATA NTERM/1/,NTER/1/
 9        WRITE(NTERM,1)
10 1      FORMAT(5X,'QUANTI SONO I PUNTI DATI DELLA FUNZIONE ?')
11        READ(NTERM,2) M
12 2      FORMAT(I4)
13        WRITE(NTERM,3)
14 3      FORMAT(5X,'FORNIRE I VALORI DI X IN COLONNA')
15        READ(NTERM,4) (X(L),L=1,M)
16 4      FORMAT(F18.8)
17        WRITE(NTERM,5)
18 5      FORMAT(5X,'FORNIRE I VALORI DI Y IN COLONNA')
19        READ(NTERM,4) (Y(L),L=1,M)
20 C
21 C      SEQUENZA DI CORREZIONE DEI DATI DELLA FUNZIONE
22 C
23 61     WRITE(NTERM,7)
24 7      FORMAT(5X,'PER CORREZIONI 0, ALTRIMENTI 1')
25        READ(NTERM,2) KOR
26        IF(KOR) 20,9,20
27 9      WRITE(NTERM,10)
28 10     FORMAT(5X,'INDICARE NUMERO ORDINE DATO INESATTO')
29        READ(NTERM,2) NOR
30        WRITE(NTERM,12) NOR
31 12     FORMAT(5X,'INDICARE IN COLONNA VALORI X,Y DEL PUNTO',I4)
32        READ(NTERM,4) X(NOR),Y(NOR)
33        GOTO 61
34 C
35 C      INTRODUZIONE E CONTROLLO DELLA X DI INTERPOLAZIONE
36 C
37 20     WRITE(NTERM,21)
38 21     FORMAT(5X,'INDICARE X DI INTERPOLAZIONE')
39        READ(NTERM,4) XS
40        DO 25 KK=1,M
41        IF(XS.EQ.X(KK)) GOTO 50
42 25     CONTINUE
43 C
44 C      APPLIC[illegible]NE DELLA FORMULA DI INTERPOLAZIONE DI LAGRANGE
45 C
46        C=1.
47        DO 30 JJ=1,M
48        C=C*(XS-X(JJ))
49 30     CONTINUE
50        YS=0.
51        DO 40 I=1,M
52        T=C*Y(I)/(XS-X(I))
53        DO 35 J=1,M
54        IF(J.EQ.I) GOTO 35
55        T=T/(X(I)-X(J))
56 35     CONTINUE

57        YS=YS+T
58 40     CONTINUE
59        GOTO 51
60 C
61 C      OUTPUT COORDINATE PUNTO INTERPOLAZIONE
62 C
63 50     YS=Y(KK)
64 51     WRITE(NTER,41)XS,YS
65 41     FORMAT(5X,'X =',F18.8,/5X,'Y =',F18.8,//)
66        WRITE(NTERM,42)
67 42     FORMAT(5X,'PER CONTINUARE 1, ALTRIMENTI 0')
68        READ(NTERM,2) MORE
69        IF(MORE) 20,44,20
70 44     STOP
71        END


 2        SUBROUTINE LAGRAN (X,Y,M,XS,YS)
 3 C
 4 C      X = VETTORE DELLE ASCISSE DEI PUNTI DATI DELLA FUNZIONE
 5 C      Y = VETTORE DELLE ORDINATE DEI PUNTI DATI DELLA FUNZIONE
 6 C      M = NUMERO DI PUNTI DATI
 7 C      XS = ASCISSA DEL PUNTO DI INTERPOLAZIONE (IN INGRESSO)
 8 C      YS = ORDINATA DEL PUNTO DI INTERPOLAZIONE (IN USCITA)
 9 C
10        DIMENSION X(M),Y(M)
11        DO 25 KK = 1,M
12        IF(XS.EQ.X(KK)) GOTO 50
13 25     CONTINUE
14        C = 1.
15        DO 30 JJ = 1,M
16        C = C*(XS-X(JJ))
17 30     CONTINUE
18        YS = 0.
19        DO 40 I = 1,M
20        T = C*Y(I)/(XS-X(I))
21        DO 35 J = 1,M
22        IF(J.EQ.I) GOTO 35
23        T = T/(X(I)-X(J))
24 35     CONTINUE
25        YS = YS+T
26 40     CONTINUE
27        GOTO 51
28 50     YS = Y(KK)
29 51     RETURN
30        END
```

Un Personal Computer Club di analisti chimici: benvenuto!

La Società Italiana Di Biochimica Clinica ha costituito un "Personal Computer Club" per promuovere l'applicazione dei piccoli elaboratori elettronici al laboratorio di analisi cliniche. Siamo interessati a programmi di matematica, statistica, grafica, gestione del laboratorio, controllo di qualità.
A tutti coloro che si mettono in contatto con il club offriamo gratuitamente un programma per il controllo di qualità in laboratorio, messo a punto da un laboratorio universitario australiano (per Apple II).
Preghiamo di segnalarci i vostri programmi e la vostra disponibilità a scambiarli con altro software. L'adesione al club è completamente gratuita.

Norberto Montalbetti, direttore del Notiziario della S.I.Bio.C.

Gli interessati si mettano in contatto con:
Personal Computer Club S.I.Bio.C.
Via Keplero, 10
20124 Milano
Tel. 02/6083042

Soccorso DAIsta (softwaroteca di 2000 titoli!)

Ho letto sul numero di febbraio 1983 di **Bit** il disperato appello di un lettore di Genova, possessore come me di un DAI PC e vorrei aiutare questo DAIsta in pena.
Il problema più grosso che abbiamo noi DAIsti sparsi per l'Italia è la grande mancanza di informazioni in merito al nostro "piccolo" ma potenzialmente potentissimo calcolatore.
Questa mancanza di notizie è da imputare soprattutto al disinteresse con cui da noi è trattato il nostro PC e non ad una reale mancanza di dati tecnici e di software prodotto per il DAI.
Esistono infatti un gran numero di club dedicati al DAI PC sparsi per tutta Europa che fanno tutti capo al DAInamic User Club belga.
Questo club pubblica una rivista bimestrale di oltre 100 pagine dedicata esclusivamente al DAI. Tutti gli interessati possono scrivere direttamente a: DAInamic User Club - Bruno Van Rompaey - Bovenbosstraat 4 - B 3044 Haasrode - Belgie - oppure anche a me che provvederò a dare tutti i consigli e le informazioni desiderate.
Posso preannunciare che sono disponibili oltre 2000 titoli, di tutti i generi, dai giochi come un Sargon 6, un Invaders, od un labirinto 3d fino ai package tecnici od amministrativi.
Per l'hardware vi sono i nuovi doppi floppydisck da 1,6 Mbit, un Eprom programmer, e schede per espansioni di memoria a lotti di 16 - 32 Kbit su RAM o Static Ram, Interfacce analogico-digitali, un ricevitore di carte meteo da satellite, una scheda per collegarsi via modem a banche di dati, e molto altro.
Vi ringrazio per l'ospitalità (seppur troppo saltuariamente) che la vostra rivista offre al DAI.

Geom. Roberto Porta
C.so F. Cavallotti, 27
15100 Alessandria

Programmazione aziendale: non tutte rose e fiori

Lavoro in un grande centro elaborazione dati, benchè forse molti hobbisti possono individuare il mio lavoro, diversi sono i punti negativi sul piano della soddisfazione personale:

— impossibilità o quasi di dedicarsi a temi d'interesse personale, che non collimino con obiettivi aziendali;
— condizionamento nella produzione del software, ispirati alla standardizzazione, che tarpano le ali alla fantasia e creatività personali;
— scarsa o nulla attenzione ai problemi di presentazione di dati e risultati, con privilegio del "data crunching" l'incessante manipolazione dei numeri e delle informazioni;
— scarso tempo dedicato alla produzione di strumenti di facilitazione del lavoro del programmatore.

È ovvio che a questi inconvenienti, peraltro correttamente inquadrabili nell'ottica aziendale, fanno riscontro una serie di gratificazioni che vanno dall'immersione in ambienti ricchi di cultura informatica e dall'ampio ventaglio di problematiche, per toccare la possibilità di sperimentare in anteprima le novità tecnologiche ed i contatti diretti con le case costruttrici.
È mia convinzione che i primi si pongano nel-l'ordine d'idee di aiutare i secondi per promuovere in ogni dove la cultura informatica. I possibili campi d'intervento potrebbero essere quelli della scelta dell'hardware e del software, dei problemi di interfacciamento o di organizzazione dei dati.
Nel ringraziarla per l'ospitalità le invio i miei complimenti per la bella rivista: vista agli inizi con distacco ed ironia negli ambienti della "grande" informatica", adesso circola sempre più sui tavoli e talora viene citata come punto di riferimento.

Bernardino Pisicchio - Martina Franca (Taranto)

Fa piacere constatare che una mentalità così aperta si vada facendo strada nei sancta sanctorum dei centri tradizionali. Detto con franchezza non tutti i segnali inducono all'ottimismo e non soltanto i due mondi restano piuttosto separati (diversità di sistemi operativi e linguaggi, differenza delle problematiche, per dire solamente le cose più ovvie) ma a quanto mi risulta si guardano in cagnesco in quegli ambienti di lavoro in cui cominciano a convivere. Mi spiego, ripetendo quanto già detto in altra occasione: in molte aziende stanno entrando i personal computer, in modo subdolo e/o selvaggio, presso utilizzatori cosiddetti finali e tale fenomeno (che, detto en passant, ha poco a che vedere col mondo hobbistico cui il nostro lettore fa esclusivo riferimento) lungi dal rappresentare l'inizio d'un idillio tra programmatori professionisti e non, in genere rinfocola e fa esplodere vecchi e mai sopiti rancori tra "sacerdoti" che temono di vedersi sfuggire un certo potere aziendale e quadri di vario tipo (specie tecnici) che snobbano in modo fin impietoso ed ingiusto esperti di informatica da loro accusati di non aver mai saputo fornire soluzioni adeguate ai loro reali problemi.

Di fronte a questi aspetti poco simpatici che parlano di "incompatibilità" di mentalità e caratteri prima ed oltre che di tipo tecnico la nostra rivista intende porsi in modo equilibrato e neutrale. Benvenuti siano quindi questi propositi dialogici: è di una coscienza professionale non disgiunta da un minimo di umiltà e di simpatia per gli altri che tutti hanno nel mondo d'oggi bisogno. A proposito, visto che non pochi tra gli informatici, chiamiamoli così (provvisoriamente) "tradizionali" sono iscritti all'AICA, Associazione Italiana di Calcolo Automatico, perchè costoro - oltre a procurarsi magari un personal computer? - non danno vita a quel gruppo d'interesse per la microinformatica che chi scrive inutilmente (e velleitariamente, per limiti personali, lo ammetto!) tentò tempo addietro di far nascere?

Personal, mondo analogico e banda passante

Sono un futuro possessore di personal computer e desidererei tanto che mi illustraste acuni dei possibili utilizzi delle porte di I/O analogiche. È possibile con esse trasformare il personal in un oscilloscopio (oltretutto a memoria)? Se si, come la mettiamo con la banda passante?

Guido Previde - Pavia

Tutto il vastissimo campo dell'acquisizione dati analogici e loro elaborazione ai fini più svariati, controlli automatici inclusi, si apre dinnanzi a chi possiede un personal dotato di ingressi analogici, che cioè incorporino internamente dispositivi di conversione analogico-digitali. In caso contrario si può sopperire con interfacce esterne che oggi non è difficile anche farsi da sè, grazie alla disponibilità di integrati d'ogni tipo. Si tratta di materia che purtroppo, a causa della crescente predominanza del software applicativo, viene trattato sempre meno su riviste come la nostra e che per giunta richiede uno specialismo non difficile ma alquanto stretto, più legato al linguaggio macchina ed ai microprocessori che non ai personal.
In ogni modo per gli usi di monitoraggio di fenomeni rapidamente variabili, ossia di banda passante un poco ampia, con i personal comuni la mettiamo piuttosto maluccio. I micro che adottano sono pilotati a non troppi Megahertz ed affidare ad un programma anche in linguaggio assoluto il compito di leggere campioni del segnale in arrivo rischia di perderne parecchi utili (ed indispensabili in base alla Teoria del Sampling che esige una frequenza di campionatura pari almeno al doppio di quella massima contenuta nel segnale). Così, su due piedi, posso pensare all'opportunità di un dispositivo che, adottando la tecnica del DMA (Direct Memory Access) riempia velocissimamente la RAM di campioni, bypassando la CPU. Dopodichè questa potrebbe "con comodo" fare il display sul monitor persino con un lento programmino in BASIC.
Se tra i lettori c'è qualcuno che se la sente di dire qualcosa di più, avendo addirittura realizzato qualcosa del genere, batta tre colpi.

Ditelo con i flow (tell it with flows, ovvero: noi microinformatici ci parliamo così)

Renato Verdiani - Castelfiorentino (Firenze)

Il diagrammino sopra riportato è la replica che il nostro simpatico collaboratore ci ha spedito, a seguito di una nostra (benevola) osservazione sulla permalosità e lo spirito polemico dei bene-male-detti toscani.
No comment, by the way, come dicono i presidenti USA. Semmai agli Etruscologi il compito di chiosare i pocheati oscuri del complessivamente eloquente documento missivo dell'Era Terza della microinformatica.

PICCOLI ANNUNCI

La rubrica è aperta a tutti i lettori. Le inserzioni sono completamente gratuite. I testi degli annunci devono essere inviati utilizzando l'apposito tagliando riportato in fondo alla rivista. Chi volesse evidenziare il proprio annuncio, con il tratteggio (■■■■) ai lati, deve unire l'importo di L. 5.000 anche in francobolli.

Bit BORSA

LO SPAZIO DEL RIVENDITORE

I rivenditori possono inserire i loro annunci all'interno di questa rubrica a condizioni particolari. Le tariffe e gli spazi disponibili sono forniti dalla Concessionaria di pubblicità Reina s.r.l. - Via Washington, 50 - 20149 Milano - Tel. (02) 4988066 - 4988067 - 4988068 - 4988069 dietro semplice richiesta, anche telefonica.

Hardware

Vendo preferibilmente zona Bologna **ZX81** nuovissimo (20 giorni di vita) + alim. + esp. 16 Kbyte a L. 400.000, regalo 5 cassette originali Rebit di cui una con 7 programmi (valore L. 1.000.000)
Stefano Calcaterra - Via Marconi, 34/2 - 40122 Bologna - Tel. 051/521063

Vendo **Sharp PC 1500** + stampante - plotter + 8 Kbyte memoria + penne, carta, ricaricatore batterie, manuali, valigetta; tutto perfetto L. 999.000 (solo per contatti personali).
Camillo Trevisan - Via A. Tassello, 26 - 35043 Monselice (PD) - Tel. 049/72115 (ore pasti)

Vendo **Sharp MZ80-K** con espansione 48 Kbyte + BASIC + super BASIC + Pascal + Assembler + cont. generale su disco + progr. finanza + progr. scacchi a 7 livelli + 60 progr. vari - per passaggio a sistema superiore vendo a L. 1.400.000.
Alberto Luvisetto - Via Boccaccio, 8 - 36015 Schio (VI) - Tel. 0445/24936

Vendo **CBM 3032** con floppy computhing 1 mega stampante 2022 interfaccia RS 232 e parallela modem acustico cassetta 150 programmi al miglior offerente.
Pier Giorgio Perron Cabus - C.so S. Martino, 4 - 10122 Torino - Tel. 011/530656

Vendo **VIC 20** Commodore usato pochissimo, più scheda espansione 8 Kbyte + grafica alta risoluzione + programmi giochi + accessori. Il tutto a L. 520.000 + L. 40.000 scheda allungagio Giove.
Bells - Via Zanotti, 21 - 40137 Bologna - Tel. 051/303098

Vendo **HP-85A** + 16 Kbyte RAM + ROM prog. avanzata + ROM mass storage + ROM printer-/plotter completo di tutti i manuali + 2 rotoli carta termica. Stampante Centronics 737 possibilità foglio singolo modulo continuo e roll-paper con manuale.
Studio Tecnico Impianti - Via Coghetti, 9 - 47037 Rimini (FO) - Tel. 0541/84000

Compro **vecchia stampante** PC100A Texas Instruments solo se a prezzo modico.
Mauro Provezza - Via Ozanam, 15 - 20129 Milano - Tel. 02/226248 (ore pasti)

Vendo **PICO2** + tastiera + interfaccia registratore + interfaccia videografica (256x256 punti) + alimentatore originale + piastra madre il tutto funzionante a L. 480.000 trattabili. Regalo inoltre 2 EPROM originali per la gestione completa del video.
Enrico Lazzerini - Via S. Agostino, 309 - 56100 Pisa - Tel. 050/42761 (ore pasti)

Vendo **ZX81** con 16 Kbyte RAM + alimentatore + manuali inglese e italiano e cassetta con 15 programmi + un flexidisc con programma il tutto per L. 400.000 inoltre per L. 100.000 vendo un CB 40 canali + antenna e alimentatore.
Angelo Vianello - Via S. Croce, 2158 - 30100 Venezia - Tel. 041/25723

Vendo **Apple II** 48 Kbyte più scheda Apple Language, Pascal, Fortran, DOS 3.3, data base L. 2.000.000.
Pasquale Leone - Via G. Santacroce, 25 - 80129 Napoli - Tel. 081/379092 (ore 21-22)

Vendo **ZX81** alimentatore, cavetti, manuali, *perfetto funzionamento, come nuovo*. Videogioco "nuova elettronica" LX446, colori, suoni, cloche e tastiere, giochi space invaders, labirinto, master mind, perfetto funzionamento. Telefonare martedì e venerdì ore serali a:
Ezio Ferraudo - Via Moncenisio, 63 - 10050 S. Antonino di Susa (TO) - Tel. 011/9640190

Lo spazio del rivenditore

ISTITUTO PADOVA
Via C. Arici, 15/a - 20127 Milano - Tel. 2566929-2563168
Per un rapido inserimento nel mondo del lavoro; per una più elevata professionalità **frequenta i nostri corsi di Microinformatica.**
Corsi teorico-pratici di:
1) Programmazione in **BASIC**
2) Programmazione in **PASCAL**
3) Corsi per operatori di immissione dati.
Per informazioni ed iscrizioni rivolgersi in sede.

Vendo **VIC 20** + mother board a 8 connettori (anche da sola) + 3 Kbyte grafica + macchine code monitor e jelly monsters (ROM) + VIC revealed + programmatore di EPROM (2716-2732-2764) + altri manuali e riviste inglesi + 200 programmi su cassette.
Leonardo Fei - Via A. Fava, 6 - 20125 Milano - Tel. 02/6894142

Vendo **Computer NE**: interf. video L. 110.000 - interf. cass. L. 80.000 - interf. esad. contast. L. 80.000 - monitor 12" verde con mob. per tastiera e monit. E1 floppy L. 180.000.
Nicola Piemontese - Via Del Palazzaccio, 49 - 50141 Firenze - Tel. 450690

Compro **VIC 20 - ZX81**
Alfredo Liberatore - Via Notarbartolo, 13A - 90143 Palermo

Vendo **VIC 20** Commodore nuovissima con interfaccia per unità a nastro - una settimana di vita - in garanzia e imballo originale L. 600.000 trattabili + 1 cassetta gioco in omaggio.
Giancarlo Andolfi - C.so Porta Ticinese, 60 - 20123 Milano - Tel. 8327856

Vendo **Sinclair ZX80** + alimentatore + suoneria + interfaccia artigianale con dispositivo di inversione di schermo a L. 220.000. Cerco inoltre VIC 20 preferibilmente zona Venezia.
Claudio Trevisan - Calseselle di S. Pietro, 11 - 30030 Oriago di Mira (VE) - Tel. 041/428422

Vendo **DAI 48 Kbyte** ancora in garanzia completo di manuali, cassette programmi e processore aritmetico (anche a parte) a L. 1.400.000 (PC) + 200.000 (AMD 9511).
Lorenzo Ambri - Via Guido Guerra, 12 - 50126 Firenze - Tel. 580940

Vendo **stampante per HP41C/CV** perfetta, 1 mese di vita, a prezzo davvero interessante.
Amedeo Fasano - Via Res. Sagittario - 20090 Segrate (MI) - Tel. 02/2130331

Vendo **Sinclair ZX-81** (1-K di RAM) + alimentatore + cavetti per registratore e televisore + manuali in italiano e inglese. Usato solo pochi mesi. Il tutto a L. 180.000
Luigi Giovagnoli - Via Di Mezzo, 19 - Villa Verucchio (FO) - Tel. 0541/678638

Vendo Commodore **VIC-20** ottimo stato L. 48.000 trattabili area Roma o Bologna. Vendo anche, assieme o separati, accessori VIC-20 e libri italiano e inglese su VIC e su microprocessore 6502 ecc.
Alessandro Martelli - Via Tiberio Imperatore, 45 - 00145 Roma - Tel. (casa) 06/5140606 - (ufficio) 06/54864732

Vendo **TI 99/4A** interfaccia cassette con alcuni programmi, joysticks; ancora imballato, con garanzia da datare, a sole L. 630.000. PET 2001, 9 Kbyte RAM, C2N, video, tutto in discreto stato a L. 500.000. Vendo solo per Torino e provincia.
Danilo Galgani - P.zza Mattirolo, 8 - 10149 Torino - Tel. 011/298053

Compro **Sharp MZ80** se vera occasione: anche modello 20 Kbyte RAM.
Elio - Tel. 06/36882788 (feriali ore 14)

Vendo **Video Genie 3003** 16 Kbyte di RAM, BASIC esteso livello II del TRS80, registratore incorporato. Numerosi programmi di giochi vari e matematici su cassette libro di programmazione in italiano più istruzioni per assembler, tutto L. 650.000.
Ermenegildo Crippa - Via Milano, 7/2 - 22050 Lomagna (CO) - Tel. (casa) 039/58345 oppure 02/741390

Vendo Hewlett-Packard **HP-41CV** + stampante 82143A, tutto in garanzia a L. 700.000
Saverio Marconi - Via Mentana, 92 - 43100 Parma - Tel. 0521/38789

Compro **EX81** 1 Kbyte o 16 Kbyte. Vendo Texas SR 56 L. 80.000.
Alessandro Giolitti - Via Fabroni, 45 - 50134 Firenze - Tel. 055/473810

Vendo **Sinclair ZX81** 16 Kbyte RAM + circa 20 programmi originali tradotti + cavetti, manuale inglese e italiano acquistato a giugno vendo a L. 350.000 trattabili. Telefonare ore pasti a:
Riccardo Arvat - Via Boston, 34 - 10137 Torino - Tel. 011/354030

Vendo **schede computer NE** LX383 interfaccia tast. L. 50.000; LX384 tastiera esadec. L. 50.000; LX385 int. cassette L. 100.000; LX388 int. video L. 200.000; LX389 int. stampante L. 50.000; perfettamente funzionanti.
Achille Ghilotti - Via Alpini, 1 - 23033 Grosio (SO) - Tel. 0342/845183

Vendo **VIC 20** + toolkit + M. monitor + sup. exp + registratore + joystick + manuali + VIC revealed + 30 programmi tra cui othello, 3-D maze ed altri introvabili (in ML) a L. 900.000 trattabili (il valore totale supera il milione). Possibilità di accordo telefonico.
Gianluca Orlando - Via Longhi, 8 - 35100 Padova - Tel. 605904

Vendo personal computer **TRS 80 I** 16 Kbyte di memoria, sistema operativo + interprete BASIC + unità a cassette. Perfettamente funzionante L. 900.000 trattabili.
Carlo Copat - Via G.B. Cevasco, 86 - 16021 Genova - Bargagli - Tel. 010/900577

Vendo a L. 20.000 **interfaccia TC2023** per collegare VIC 20 con qualunque registratore. Vendo per VIC 20 "caccia all'anatra" "breakout" a L. 15.000 ognuno. Vendo programmi per ZX Spectrum, ZX81, Atari, BBC, dragon. Ogni listato L. 5.000. Consulenza acquisti di computer.
Andrea Faraboli - Via Del Parco, 38 - 44100 Ferrara

Vendo **Sharp MZ80K** nuovo 48 Kbyte RAM completo di interfaccia stampante parallela e software: BASIC - machine language - assembler L. 1.500.000.
Maurizio Vanini - Via Scrimei - 37013 Caprino Veronese (VR) - Tel. 045/7241539

Vendo affare **MZ80A** 48 Kbyte L. 1.750.000 ed MZ80B L. 3.170.000 - Nuovi imballati prezzo tutto compreso, forniamo inoltre programmi gestionali in BASIC e CP/M.
Rizzi Davide - Via Falck, 19 - 20151 Milano - Tel. 02/3531778

Software

Vendo **interprete BASIC** 8 Kbyte su cssetta per micro Z80 NE. Comandi e funzioni facilmente espandibili. Documentato L. 60.000.
Massimo Pierazzuoli - Via Medardo Rosso, 25 - 50142 Firenze - Tel. 055/700558 (ore 18/22)

Vendo per il DAI 48 Kbyte fantastici **programmi** garantiti inediti, con colore e suono. Savefight (invasori) su cassette L. 14.000; reverse su cass. L. 14.000. 10 programmi su cassetta (invasori, midway, ecc.) L. 20.000. Contrassegno L. 1.700 in più. Dispongo anche centinaia di programmi per Sinclair a prezzi bassissimi. Per informazioni allegare francobolli.
Vincenza Avena - Via Garibaldi - 04016 Sabaudia (LT)

Cambio **software** personal computing di Triumph Adler Alphatronic P2. Sono in possesso di giochi programmi di calcolo matematico e finanziario di utilità - disposto a cambiare o a comprare anche solo listati.
Antonio Masala - Via Tempio, 39 - 07100 Sassari - Tel. 079/272663

Vendo **programmi di geometria** e matematica per HP-41 C/CV adatti per studenti in ingegneria e matematica.
Michele Magni - Via Mameli, 15 - 21100 varese - Tel. 0332/239557

■ *Vendo* **software ingegneria** civile per M20 ST: calcoli statici e verifica telai piani - analisi sismica - correzione torsionale e termica - relazione tecnica - rogetto solai, balconi, scale.(Zona di Messina rivolgersi a "CS80" - Tel. 2924641)
Ing. Mariano Rossello - Via Salbertrand, 76 - 10146 Torino - Yel. 011/756421-756521 ■

Studenti in informatica *vendono* **programmi** in BASIC, Pascal e Fortran di ogni tipo pronti o su richiesta per Apple II e III. Analisi e consulenze per l'acquisto di sistemi per l'elaborazione dati di qualsiasi dimensione.
Giampaolo Britti - Via Ascanio Fenizi, 52 - 00149 Roma - Tel. 06/55704130 (ore pomeridiane)

Ho 11 anni e possiedo un TI-99/4A Texas, mi interessano **programmi** listati o su cassetta.
Simona Gallina - Via G. Miel, 3 - 10023 Chieri (TO)

Vendo eccezionali **giochi** per VIC 20 su cassetta per configurazione di base o con espansione 8 Kbyte.
Alessandro Sicoli - Via Ciccotti, 8 - Milano - Tel. 02/6466115 (ore pasti)

Vendo **software ZX81** è pronto un programma per fatturazione, preventivi, conti, listino prezzi e articoli, personalizzabile. L. 30.000 in cassetta.
Ivano Pongiluppi - Via Roosevelt, 63 - 41012 Carpi (MO) - Tel. 059/683923 (ore pasti)

Vendo **programma per diagnosi medica** su Apple II. Consente a tutti gli effetti la diagnostica medicale computerizzata.
Pietro Budicin - Casella Postale 758 - 34100 Trieste - Tel. 040/910300

Cambio **software** applicativo gestionale sotto CP/M per micro NE.
Angelo Lamon - Via Orlanda, 178/B - 30030 Campalto (VE) - Tel. 041/903531

Compro/Cambio **programmi HP 41C/Y**. Cerco ottimo software per ingegneria civile e meccanica. Cambio buoni programmi di ingegneria civile con altrettanti ben documentati.
Fortunato Palermo - C.so Vittorio Emanuele II, 57 - 10128 Torino - Tel. 011/535724

Compro pagando bene, **programmi per Atari 400** sia di utility che di game.
Andrea Verona - Via Masceroni, 12 - 20149 Milano - Tel. 02/495814

Neopossessore pocker computer Sharp PC 1500 inesperto programmazione ***cerca* programmi** qualsiasi genere, in particolare scientifici e tecnici per suddetto computer.
Maurizio De Gioanni - C.so S. Santarosa, 67 - 12100 Cuneo - Tel. 0171/61834

Vendo **programmi ZX81** inediti su cassetta grafica eccezionale: banco, l'antico maniero, derby, allunaggio, ZX pack man, black jack ecc. - elenco programmi disponibile su richiesta a L. 1.000 per spese spedizione.
Maurizio Laurenti - Via Emanuele Filiberto, 257 - 00185 Roma - Tel. 06/7575716

Vendo **programmi novità** divertenti anche didattici per ZX81. Bombardiere, mastermind, black jack, su cassetta da L. 5.000 a L. 8.000. Stampati su carta L. 2.000. Vastissimo assortimento!! Telefonare dopo le ore 20 a:
Giuseppe Morelli - Piazza Salvatore Galgano, 90 - 00173 Roma - Tel. 7483700

Vendo **eccezionali programmi** per il VIC 20 tutti a metà prezzo di listino. Telefonare ore pasti a:
Florindo Palladino - Via Bovisasca, 161 - 20161 Milano - Tel. 02/3550858

Vendo per VIC 20 (3-8 Kbyte) **bellissimi giochi** e utilities (word-processing ed altre) di alta qualità su cassetta a partire da L. 8.000 a L. 35.000. A richiesta si invia ampia documentazione (allegare L. 1.000). Scrivere a:
Carlo Zanini - Casella Postale 8 - 26100 Cremona - Tel. 0372/27498

Vendo **software civile** Pet ZX81 completissimo supertestato verifica sezioni, telaio, grigliati, fondazioni, muti, zona sismica, 372, equo canone ecc.
Giovanni Gaviani - Via Finelli, 3 - Bologna - Tel. 051/230126

Vendo **listati** di programmi per VIC 20, ZX81, ZX spectrum, Atari. Ogni listato L. 5.000. Siamo tre studenti di ingegneria elett. e offriamo la nostra esperienza, dietro modesto compenso, a persone nuove del settore che vogliono comperare un PC.
Andrea Faraboli - Via Del Parco, 38 - 44100 Ferrara

Vendo **programma** completo per CBM 3032 Commodore relativo alla redazione della relazione geotecnica di calcolo dei cedimenti di fondazioni superficiali poggianti su terreni stratificati - versione video o stampa.
Ing. Alvaro Albani - Via Castelfidardo, 17 - 47037 Rimini (FO) - Tel. 25765-55350

Cerco **programmi per TI 99/4A** della Texas Instruments disposto anche a pagare. Telefonare ore pasti a:
Giovanni Toscanini - Via Catlinetti, 17 - 13010 Isolella Sesia (VC) - Tel. 0163/22473

Vendo **trenta programmi** per lo ZX81 non espanso al modico prezzo di sole L. 9.000. I programmi vengono venduti già registrati su cassetta inclusa nel prezzo e non richiedono alcuna espansione RAM.
Livio - Fabio Pomi - Via B. Giacomini, 2 - 21051 Arcisate (VA) - Tel. 0332/470343

Cambio **programmi** per Apple II di qualsiasi genere: giochi, utility, gestionali, vari.
Vincenzo Izzo - Via Caldieri, 143 - 80128 Napoli - Tel. 081/648219

Cerco **possessori PET/CBM** per scambio programmi di vario genere. Se interessati inviare lista o telefonare. Cerco anche nella zona di Firenze città possessori PET. Rispondo a tutti.
Andrea Chiuppi - Via Diocleziano, 41 - 33010 Feletto Umberto (UD) - Tel. 0432/681479

Vendo **guerre stellari** per ZX81/16 Kbyte, superprogramma galattico a L. 5.000.
Luciano Bellotto - Via S. Pietro, 10 - 10034 Chivasso - Tel. 9111219

Cambio, compro, vendo **programmi** per Apple II. Cerco language card da scambiare con programmi.
Claudio Citarella - Via Parroco Federico, 41 - 80045 Pompei - Tel. 081/8632946

Compro **programmi** di ogni genere per ZX 80 con memoria da 1 Kbyte. Telefonare ore pasti o scrivere a:
Marco Nascé - Via Volturno, 3 - 30173 Mestre (VE) - Tel. 041/980968

Vuoi scambiare **software per ZX80** ZX80N, ZX81 con me? Mettiti in contatto - cambio software di qualsiasi tipo, vuoi programmi o routine in linguaggio macchina o in BASIC inoltre scambi di idee e di consigli su nuovi programmi, ecc.
Giuseppe Monteleone - Via Monzoro, 20 - 20010 Cornaredo (MI) - Tel. 02/9362908

Non dispongo più del computer Apple II e quindi vendo a prezzi irrisori e fino ad esaurimento tutti i **dischetti (giochi - ing. civile)** ancora in mio possesso. Scrivetemi, vi manderò elenco e prezzi.
Roberto Vigone - Via Attone Vescovo, 24 - 13100 Vercelli

Compro **programmi** vari per VIC 20 con o senza espansioni e/o accessori, cerco inoltre libri in lingua straniera e manuale della 3K/grafica. Contatto inoltre vickeristi provincia Savona per fondare club.
Davide Zegna - Via Marco Polo, 1 - 17025 Loano - Tel. 019/670582 (ore pasti)

Vendo **Cartridge** VIC graph e poker nuovissimi a L. 120.000. Telefonare sabato e domenica a:
Paolo Nappo - Via Vernilli - Ciommi, 36/A - 80047 S. Giuseppe Ves.no (NA) - Tel. 081/8281981

Vendiamo o al limite scambiamo e realizziamo **programmi** su commissione per Apple II e III anche in Pascal e Fortran. Telefonare dopo le ore 19 a:
P.M. Software House - Via Veneziano, 10 - 20139 Milano - Tel. 02/531505

Vendo **software per TI/99 4A**: videogames con grafica a colori e suoni, tutti perfettamente funzionanti; BASIC 12-14 Kbyte; anche in cassette; prezzi sulle L. 10.000. Disposto anche a scambiare ma solo per la zona di Milano.
Paolo Ventafridda - Via Ottoboni, 6 - 20148 Milano - Tel. 02/4032432

Vendo **programmi di ingegneria** civile per computer CBM serie 40 e 80 - risoluzione telai calcolo armature cementi armati - legge 373 - metodo por - computi metrici - revisione prezzi - analisi sismica - solai - word processor.
Studio Strctura - Via Marcona, 24 - 20129 Milano - Tel. 02/718823

Vendo o cambio **programmi** per VIC 20. Inviare L. 1.000 per documentazione e prezzo.
Marino Palazzesi - C.so Grosseto, 203 - 10147 Torino - Tel. 011/217146

Vendo per VIC 20 e Apple II **programmi** di elettrotecnica e matematica: integrali, integrali doppi, radici di equazioni, equazioni differenziali, sistemi e tanti altri. Su cassetta o dischetto sono muniti di pratici manuali ed esempi.
Luigi Racioppi - Via A. Diaz, 114 - 80055 Portici (NA)

Vendo o cambio **cartuccia per TI 99/4A** contenente: tiro a segno, flipper, domino. Grafica buona, valore reale L. 70.000 vendo a L. 50.000 oppure cambio. Compro anche programmi sempre per il TI99/4A e cerco aderenti per il Firstitalian Club of tele electronic meeting (fictem)
Marco Sarzina - Via Bellini, 17 - 25077 Roé Volciano (BS)

Vendo **cassette programmi** *giochi per HP85 (25 programmi) (pocer-blacks) ecc. L. 100.000 e per HP9845T 16 giochi L. 100.000 ho anche i programmi della users library e il Visicalc per HP85 il tutto posso cambiare con programmi per Apple II*
Natalino Doro - Via Carducci, 5 - 24100 Bergamo - Tel. 035/256998

vendo per ZX80 **programma tartinville** (per ricerca capisaldi in equazione parametrica 2° grado) a L. 15.000 tutto compreso (cassetta + spese spedizione). Vendo inoltre diversi programmi di vario genere. Per contatti scrivere a:
Bruno Cardella - Via Calabria, 4 lotto 43 - 90100 Palermo - Tel. 512302

Vendo **monitor** 9 pollici L. 5.000 generatore sweep VHF con marker canali TV L. 200.000.
Giuliano Gatti - Via Cagliero, 9 - 20125 Milano - Tel. 02/6070190

Vendo le seguenti **schede** del **micro NE** Z80: LX380, LX381, LX382, LX385, LX387, LX392. Tutte le schede sono montate e perfettamente funzionanti. Il blocco di schede costa L. 550.000 le singole schede costano un 15% in meno rispetto al prezzo del kit.
Marco Caraveo - Via Sapeto, 45A - 16132 Genova - Tel. 396500

Vendo **videogioco "Mesaton"** + 2 cartucce. Progetto di nuova elettronica, in ottimo stato ed usato poco L. 150.000. Telefonare sabato o domenica ore serali.
Paolo Geronazzo - Via Don Formentini - 21010 Bosco Montegrino (VA) - Tel. 0332/589739

Vendo per micro Z80 NE **LX389** (interfaccia stampante) L. 50.000 - LX 392 (memoria din. 32 Kbyte) L. 145.000 - scheda video CPV001 (80x24) originale mai usata L. 220.000. Il tutto a L. 400.000. Telefonare ore serali a:
Gino Di Mambro - Via Licinio Murena, 36 - 00175 Roma - Tel. 06/768392

Vendo eccezionale!! **Cartridge** monitor linguaggio macchina per VIC 20 avente 32 istruzioni, molto superiore a quello Commodore (solo 19 istruzioni completa di utilissimo tasto reset per L. 55.000)
Giovanni Torre - Via Leonardo Da Vinci, 69 - 80055 Portici (NA) - Tel. 081/481439-482706

Vendo **telescrivente** ASR-33 con perforatore-lettore di nastro, interf. seriale loop zona completa di piedistallo, alim. 220 V, revisionata L. 550.000. Telefonare dopo le ore 18 a:
Randolfo Basile - Via Mattei, 14 - 20018 Sedriano (MI) - Tel. 9020725

Cambio o vendo **organo Bontempi** HF201 6 ritmi combinabili 42 tasti 3 toni, nuovo per doppio regalo L. 250.000 oppure cambio con ZX81 o simili.
Roberto Borghi - Via G. Verdi, 10 - 46026 Quistello (MN) - Tel. 0376/618016

Vendo microcalcolatore **Chess Champion MK 1** 6 livelli di abilità, soluzione problemi, tutte le mosse del gioco sono impostabili usato pochissimo, come nuovo L. 100.000
Giulio Piovene - Via G. Prati, 13 - 35100 Padova - Tel. 049/656153

Vendo **scheda grafica** nuova elettronica con beep (LX529+530) perfettamente montate e funzionanti a L. 300.000.
Vincenzo Vitale - Via N. Nicolini, 27 - 66100 Chieti - Tel. 0871/63139

Vendo **frequenzimetro LX 358** di nuova elettronica, mai usato e perfettamente funzionante L. 250.000. Telefonare il sabato sera o domenica a:
Paolo Geronazzo - Via Don Formentini - 21010 Montegrino Bosco (VA) - Tel. 0332/589739

Vendo stampante **Teletype ASR33** su piedistallo con lettore-perforatore banda cod. ASCII perfettamente funzionante. Interfaccia 20MA current loop. Prezzo L. 300.000 non trattabili.
Max Salvi - Via S.G. Bosco, 12 - 20047 Brugherio (MI) - Tel. 039/870752 (ore serali)

Per fondazione club a livello nazionale **contatterei possessori** computer **Texas TI 99/4A**. Scrivere allegando busta affrancata con indirizzo per la risposta a:
Marina Travaglini - Casella Postale 6315 - 00195 Roma Prati

Cerco **espansione** memoria e interfaccia RS232 per EG3003 Genius Computer.
Angelo Ravagli - Via Entirate, 59/A - 48020 Traversara (RA) - Tel. 0545/49053

Vendo i primi 4 **volumi di nuova elettronica** a L. 5.000 l'uno - i primi 6 numeri dell'Enc. di Scienza e Tecnica con cop. a L. 8.000 - Cambio progr. da 8 Kbyte Pet serie 3000 in particolare progr. per radio amatori - telefono senza fili Lake porata 100 m a L. 160.000
Sergio Daraghin - Via Paesana, 4 - 10042 Nichelino (TO) - Tel. 011/6272087

Vendo **videogame Atari**: console + cassetta "Combat" L. 200.000, a parte le cassette: space inv., surround, soccer a L. 30.000 l'una; asteroids L. 45.000, tratto solo nel veneto.
Marco Silvestri - Via A. Gregori, 4 - 36057 Arcugnano (VI) - Tel. 0444/550238 (ore 18.00-21.00)

Vendo **ROM system** per CBM 40/8032-3032 BASIC 4 aggiunge 16 com. al BASIC tra i quali contr. cursore-sort alfanumerico - somma - sottraz. - moltipl. in tripla precis. (24 cifre) con puntegg. automat. del numero - input controllato etc. Zoccolo ROM 9000 o A000; costo L. 200.000 con istruz.
Giuseppe Mannino - V.le Primavera, 3/6 - 16148 Genova - Tel. 010/332827

Intendo costruire un **Apple club** chi è interessato può rivolgersi a:
Luciano Saltarelli - Via Canero, 9 - 24010 Ponteranica (BG) - Tel. 035/248377

Vendo **modulo per luci rotanti** ad 8 canali in grado di pilotare lampade a 220 V - 250 W max con regolazione della velocità. Il tutto autocostruito e montato in un contenitore in legno, a L. 40.000 + spese di spedizione.
Enzo Sarcuni - Via Don Luigi Sturzo, 2 - 75100 Matera - Tel. 0835/261900

Ragazzi (max anni 20) proprietari di ZX80-81 e/o Pet cercansi per **formare un club**. Posso insegnare ad usare i computer sopra elencati. Per informazioni scrivere a:
Michele Bighignoli - Via Palermo, 40/12 - 39100 Bolzano - Tel. 0471/915107

Vendo da USA e GB, **nuovissimi libri** per utenti VIC 20: Zap Pow boom; VIC innovative comp.; Getting acquainted with VIC 20 e utenti Atari; Atari sound and graphics; your Atari comp.; Atari games and recreation.
Roberto Grandi - Via Tonale, 14 - 20030 - Bovisio (MI) - Tel. 0362/592453 - ufficio 02/88474639

Vendo **drive** con controller per floppy disk 5", per Apple, nuovo (ricevuto come premio) L. 1.000.000
Telefonare a: 02/230352

Vendo per NE Z80 **drive BASF** 5" 1/4 + interfaccia floppy disk con relativa piattina + alimentatore + NE DOS + documentazione ampia il tutto a L. 600.000 solo se veramente interessati poss. zona Roma e dintorni.
Roberto Rocchetti - Via Casilina - 00176 Roma - Tel. 06/2776324 (ore serali)

Cerco **amici Apple** zona Savona per fondazione Club. Contattare ore pasti.
Roberto Nervi - Via Famagosta, 26/4 - 17100 Savona - Tel. 019/32753

Cambio **TV games Mesaton** + Mesacomp 1 con istruzioni 8 ROM giochi - nastri - programmabile in esadecimale, con TV/C 12 - 16 pollici anche senza telecomando. Scrivere o telefonare ore serali.
Eligio Furlani - Via Bobbio, 15/5 - 16137 Genova - Tel. 816705

Sinclair computer Club; disponibili per ZX80/81 software e hardware inediti: Forth, RS232, alta risoluzione Assembler, W-processing, ecc.
Gian Luca Carri - Via Forlivese, 9 - 50065 Pontassieve (FI) - Tel. 055/8304677

Istituto **TRS80PC** - Sharp PC1211 **user Club**. Potrete vendere e/o acquistare programmi ed accessori. Possibilità di istituire giornale interno. Quota di iscrizione L. 18.000. Per informazioni allegare L. 400 in francobolli.
Umberto Cocchi - Via G. Bonito, 11 - 80129 Napoli - Tel. 081/248641

Vendo **CB Midland** 6001, 40 CH AM LSB-USB in PLL + lineare Lace 650 W in AM e 1300 in banda. Vera occasione L. 500.000 anche separatamente, o cambio con VIC 20 o ZX81 espanso. Telefonare ore pasti a:
Roberto Caire - C.so F. Cavalletti, 86 - 14100 Asti - Tel. 0141/353507

Vendo a L. 20.000 annata *completa (1982)* dell'**Apple Orchard**, la più diffusa rivista americana per l'Apple II. Numerosi programmi e articoli tecnici. Telefonare ore ufficio.
Paolo Ing. Benedetti - Via Oldoni, 1 - 13100 Vercelli - Tel. 0161/54937

BIT LIBRI

Gruppo Editoriale Jackson

Vuoi ordinare dei libri? Spedisci questo tagliando a: Gruppo Editoriale Jackson Via Rosellini, 12 - 20124 Milano.

Nome Cognome

Indirizzo

Cap. Città Provincia

Partita I.V.A. (indispensabile per le aziende)

Si richiede l'emissione della fattura ☐

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

☐ Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione:

☐ Allego assegno n° .... di L. ....

☐ Non abbonato ☐ Abbonato sconto ☐ l'Elettronica ☐ Elettronica Oggi ☐ Automazione Oggi ☐ Elektor
☐ Informatica Oggi ☐ Computerworld ☐ Bit ☐ Personal Software ☐ Strumenti Musicali ☐ Videogiochi

Data .... Firma ....

BIT LIBRI

Gruppo Editoriale Jackson

Vuoi ordinare dei libri? Spedisci questo tagliando a: Gruppo Editoriale Jackson Via Rosellini, 12 - 20124 Milano.

Nome Cognome

Indirizzo

Cap. Città Provincia

Partita I.V.A. (indispensabile per le aziende)

Si richiede l'emissione della fattura ☐

Inviatemi i seguenti libri:

Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità	Codice Libro	Quantità

☐ Pagherò al postino il prezzo indicato + L. 2.000 per contributo fisso spese di spedizione.

☐ Allego assegno n° .... di L. ....

☐ Non abbonato ☐ Abbonato sconto 10% ☐ l'Elettronica ☐ Elettronica Oggi ☐ Automazione Oggi ☐ Elektor
☐ Informatica Oggi ☐ Computerworld ☐ Bit ☐ Personal Software ☐ Strumenti Musicali ☐ Videogiochi

Data .... Firma ....

BIT BORSA

Sei un lettore di BIT e vuoi entrare in contatto con tutti gli altri lettori per comprare, cambiare o vendere il tuo sistema o qualche particolare accessorio? Spedisci questo tagliando a: Gruppo Editoriale Jackson, Via Rosellini, 12 - 20124 Milano.

☐ COMPRO ☐ CAMBIO ☐ VENDO

Nome.. Cognome ....

Via .... C.A.P. ....

Città .... Tel. ....

BIT LO SPAZIO DEL RIVENDITORE

Sei un rivenditore di personal computer e vuoi entrare in contatto più diretto con tutti i lettori di BIT? Spedisci questo tagliando a: Gruppo Editoriale Jackson, Via Rosellini, 12 - 20124 Milano

DITTA O NEGOZIO

VIA

CITTA' CAP.

TEL.

MARCHE RAPPRESENTATE

SPECIALIZZATO IN ...

ALTRE NOTIZIE

FIRMA

Apple parla la tua lingua

Indubbiamente con la sua semplicità e versatilità d'uso Apple parla un linguaggio universale. Non per niente, nel mondo sono piú di 700 mila i possessori di un personal computer Apple.

I personal Apple fanno di tutto per venire incontro alle esigenze piú personali. Così il nuovo Apple //e, che puoi trovare in oltre trecento centri di dimostrazione e vendita in Italia, non solo offre piú memoria e maggiore semplicità d'uso, ma anche una tastiera italiana.

I nuovi Apple parlano e scrivono nella tua lingua: nel lavoro avrai un partner che capisce meglio i tuoi problemi e li risolve prima; nello studio e nel tempo libero un compagno socievole con una fantasia tutta italiana.

Vieni a scambiare quattro chiacchiere con il nuovo Apple //e e con il piú potente Apple ///. Scoprirai che sono sorprendentemente semplici anche nel prezzo: Apple //e completo di video e unità per dischetti è in offerta speciale a soli ***2 milioni 850 mila*** e Apple /// ha un discorso ancora piú interessante da farti.

apple Il Personal Computer

Distribuzione unica per l'Italia: IRET Informatica S.p.A - Via Bovio, 5 - 42100 Reggio Emilia - Tel. 0522/32643 - Tlx 530173 IRETRE I